Haaga-Helia
ammattikorkeakoulu Oy

IT-testaajan arki yksityisellä sektorilla

Juuso Heikkinen

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2017

Haaga-Helia ammattikorkeakoulu Oy

Tiivistelmä

Päiväys 1.12.2017

Tekijä(t)
Juuso Heikkinen

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön otsikko
IT-testaajan arki yksityisellä sektorilla

Sivu- ja liitesivumäärä
84 + 0

Opinnäytetyön otsikko englanniksi
The weekdays of the IT-tester in the private sector

Päiväkirjaopinnäytetyö ”IT-testaajan arki yksityisellä sektorilla” kuvaa kesätyösuhdetta teleoperaattoriyritys Telian Quality Management-tiimissä kymmenen raportointiviikon ajan. Työ käsittelee kesätyösuhteen aikana tapahtunutta oppimista ja kehittymistä testiautomaatioon liittyvien työtehtävien kautta. Pohjana opinnäytetyön käynnistymiseen toimivat entuudestaan tuntematon aihealue ja kesätyösuhteen sopiva kesto opinnäytetyön vaatimuksia silmällä pitäen.

Työssä tarkastellaan työsuhteen aikana tehtyjä, testiautomaatioon liittyviä työtehtäviä. Näihin työtehtäviin kuului automaattisten testien ohjelmointi, joiden avulla testattiin kolmea eri Telian järjestelmää niiden graafisen käyttöliittymän kautta. Testattavat järjestelmät olivat Telian verkkokauppa, eräs myyntijärjestelmä ja Telian asiakkaan omat sivut. Tärkeimpinä työkaluina työtehtävissä toimivat Robot Framework ja PyCharm. Robot Framework on avainsanapohjainen testiautomaatioviitekehys automatisoitujen testien ohjelmoimiseen. Tässä työssä kuvattujen automatisoitujen testien ohjelmointi tapahtui PyCharmissa, joka on integroitu kehitysympäristö.

Viikoittaiset raporttimerkinnät kuvaavat työtehtävien etenemistä ja niistä suoriutumista päivittäisellä ja viikoittaisella tasolla. Raportointimerkinnät käsittelevät työpäiville asetettuja tavoitteita, vastaan tulleita ongelmatilanteita, havaittua kehitystä ja työssä hyödynnettyjä toimintatapoja. Raportoitavien viikkojen aikana selkeää oppimista tapahtui käytettäviin työkaluihin ja järjestelmiin liittyen. Erityisesti kehitystä tapahtui Robot Frameworkia käytettäessä. Tämä oli huomattavissa yleisen osaamisen parantuessa ja uusia toimintatapoja ja avainsanoja hyödynnettäessä.

Opinnäytetyön loppuanalyysi tutkii tarkemmin minkä tasoista oppimista ja kehitystä opinnäytetyön aikana tapahtui. Analyysi käsittelee myös minkälaisia ratkaisumalleja ja -menetelmiä tehdyissä työtehtävissä hyödynnettiin. Työsuhteen alkaessa lähes kaikki työtehtävissä tarvitut välineet ja järjestelmät olivat entuudestaan tuntemattomia, joten työsuhteen aikana erilaisiin työkaluihin ja järjestelmiin liittyvä kehitys oli huomattavaa. Kymmenen raportointiviikon jälkeen käytettävien järjestelmien ja työkalujen hallinta ja osaaminen oli parantunut selkeästi. Työskentelyssä oli myös hyödynnetty aktiivisesti uusia opittuja ratkaisumenetelmiä.

Asiasanat
Robot Framework, testiautomaatio, testaus, Telia, yksityinen sektori

Sisällys

1 Johdanto

Opinnäytetyö ”IT-testaajan arki yksityisellä sektorilla” kuvaa työskentelyä teleoperaattoriyritys Telian Quality Management-tiimissä kymmenen raportointiviikon ajan, maanantai 12. kesäkuuta 2017 ja sunnuntai 27. elokuuta 2017 välisenä aikana. Työ esittelee päivittäisellä tasolla kesätyösuhteen aikana testiautomaatioon liittyviä työtehtäviä ja niiden etenemistä. Työ tarkastelee päivittäin myös itselleni asettamia tavoitteita ja niiden toteutumista. Jokaisen raportointiviikon lopuksi analysoidaan syvemmin viikon aikana tapahtunutta oppimista, kehitystä, mahdollisia vastaan tulleita haasteita sekä työtehtävissä hyödynnettyjä toimintatapoja.

Pohjana opinnäytetyön aloittamiseen toimi mahdollisuus tutustua täysin uuteen aihealueeseen, josta minulla ei ollut aikaisempaa kokemusta tai tietämystä. Myös tästä syystä yhdeksi opinnäytetyön selkeimmistä tavoitteista nousi oman tietopohjan ja käytännön taitojen kehittäminen. En esimerkiksi ollut ennen työsuhteeni alkua kuullut Robot Frameworkista ja kaikki testaamiseen liittyvä osaamiseni koostui täysin viime keväällä suorittamani työharjoittelun aikana saamastani kokemuksesta manuaalisesta testauksesta. Kesätyösuhde Telialla oli oppimisen ja ammatillisen kehityksen kannalta erinomainen mahdollisuus, jonka pystyin dokumentoimaan opinnäytetyön muodossa. Myös kesätyösuhteen kesto oli juuri sopivan mittainen päiväkirjamuotoiselle opinnäytetyölle mikä vaikutti myös sen aloittamiseen.

1.1 Tietoperusta

Työsuhteeni alkaessa tietoperustanani toimivat vain koulussa, eri kesätyösuhteideni sekä opintoihini liittyvän työharjoitteluni aikana keräämäni tiedot ja käytännön taidot**.** Kesätyösuhteeni aikana vastaan tulleet työtehtävät vaativat laajaa tietoperustaa ja osaamista, minkä vuoksi minun oli opittava ja hallittava käytettävien työkalujen ja ohjelmistojen lisäksi myös käyttämäni ja testaamani Telian järjestelmät. Tarvittavaan tietoperustaan ja osaamiseen kuuluivat myös testattavien prosessien, kuten eri tuotteiden tilausprosessien, tunteminen. Testaamiini Telian järjestelmiin ja prosesseihin liittyvän tietoperustan sain minua kouluttaneilta Telia työntekijöiltä ja hyödyntämällä yrityksen sisäverkosta löytyviä dokumentaatioita.

Tärkein työtehtävissä tarvitsemani työkalu oli testiautomaatioviitekehys Robot Framework. Siihen liittyvän tietoperustan keräämiseksi hyödynsin varsinkin ensimmäisten viikkojen aikana Robotin kehittäjien eli Robot Framework Foundationin tekemää ja ylläpitämää dokumentaatiota Robot Framework User Guidea (2017c). Kyseinen dokumentaatio on pe-

rusteellinen käyttöohje Robotin käyttöön ja teoreettisuutensa takia myös oiva pohja myöhemmälle, muista lähteistä saadulle tietoperustalle.

Robot Frameworkin lähdekoodin ollessa avointa, siitä löytyy suuria määriä erilaisia dokumentaatiota. Tästä syystä suurin osa Robottiin liittyvästä tietoperustasta kertyikin työsuhteeni aikana muista lähteistä kuin pelkästään Robot Framework User Guidesta. Näitä muita lähteitä olivat muun muassa Robot Frameworkin sivuilta löytyvät muut dokumentaatiot sekä Stack Overflow- ja Robot Frameworkin käyttäjien Google Groups-sivustot. Näiden lähteiden lisäksi tietoperustani karttui koko työsuhteeni ajan testiautomaatiosta vastaavan kollegani tekemien esimerkkien ja antaman opetuksen kautta.

Sekä työsuhteeni aikana että sen jälkeen perehdyin myös Cem Kanerin, James Bachin ja Bret Pettichordin kirjoittamaan teokseen Lessons Learned In Software Testing (2002). Kirja esittelee kirjailijoiden omien, testaamisen eri osa-alueisiin liittyvien kokemuksien pohjalta kirjoittamia lyhyitä ”oppitunteja”. Näitä kirjailijoiden kokemuksia peilaamalla pystyin entistä tehokkaammin tarkastelemaan ja vertaamaan omia testaamiseen ja testiautomaatioon liittyviä mielipiteitäni ja toimintatapojani.

1.2 Työympäristö

Työympäristönäni kesätyösuhteeni aikana toimivat teleoperaattoriyrityksen Telian tilat Teollisuuskadulla Helsingissä. Olin osa noin parikymmenhenkistä Quality Management-tiimi, jossa työskentelin koko kesätyösuhteeni, 12 viikon ajan, ensimmäinen kesäkuuta ja 31. elokuuta välisenä aikana, Technical Specialist-nimikkeen alla. Työympäristönä Telia oli minulle entuudestaan tuntematon tämän ollessa ensimmäinen työsuhteeni kyseisessä yrityksessä. Työympäristönä Telia osoittautui kuitenkin hyväksi ystävällisten ja asiallisten kollegoiden, ammattitaitoisen esimiehen, mielekkäiden ja haastavien työtehtävien sekä hyvän työilmapiirin ansiosta.

1.3 Ammattikäsitelista

Tässä kappaleessa listaan kaikki tärkeimmät ja keskeisimmät ammattikäsitteet, joihin törmäsin työsuhteeni aikana:

CRM-järjestelmä	Customer Relationship Management- eli asiakuudenhallintajärjestelmä.
Deveo	Ilmainen, graafisen web-käyttöliittymän omaava versionhallintatyökalu.

GitHub	Ilmainen versionhallintatyökalu.
HTML	HyperText Markup Language eli kuvauskieli elektronisten dokumenttien, internetsivujen luomiseksi (Computer Hope, 2017).
IDE	Integrated Development Environment eli integroitu kehitysympäristö, esimerkiksi PyCharm (JetBrains, 2017a).
Intranet	Yrityksen sisäverkko.
JavaScript	Kevyt ja monialustainen ohjelmointikieli (Mozilla Foundation, 2017b).
Jenkins	Jatkuvaan integraatioon tarkoitettu applikaatio testien automaattiseen ajamiseen (Shatzer, 2016).
PyCharm	Integroitu kehitysympäristö Python-ohjelmointiin (JetBrains, 2017a).
Omat sivut	Telian järjestelmä, jonka kautta Telian asiakkaat voivat hallinnoida muun muassa omia liittymiään.
Python	Ohjelmointikieli, kuten Java ja C#.
Robot Framework	Avainsanapohjainen testiautomaatioviitekehys (Robot Framework Foundation, 2017b).
Myyntijärjestelmä	Eräs Telian myyntijärjestelmä Telian työntekijöille.
Tiketti	Tehtävänhallintajärjestelmään, kuten Jiraan, kirjattu tehtävä.
Verkkokauppa	Telian verkkokauppa, josta asiakas voi tilata Telian tuotteita tai palveluita.
VPN	Virtual Private Network eli virtuaalinen erillisverkko esimerkiksi oman IP-osoitteen piilottamiseksi (ExpressVPN, 2017).

Web-elementti	Internetsivun html-koodista löytyvä elementti
XPath	XML Path Language eli kieli, jonka avulla pystytään liikkumaan ja etsimään elementtejä XML-dokumenteista (W3Schools, 2017c).

2 Lähtötilanteen kuvaus

2.1 Oman nykyisen työn analyysi

Koska työni toimenkuva ja työtehtävissä käytettävät järjestelmät ja työkalut olivat minulle täysin entuudestaan tuntemattomia, osaamiseni taso oli näihin liittyen töiden alkaessa alhainen. Ensimmäisen viikon aikana työpäiväni sisälsivät lähes pelkästään erilaisia perehdytyksiä käytettäviin järjestelmiin ja työkaluihin liittyen. Nämä perehdytykset jatkuivat ja syventyivät vielä muutaman seuraavan viikon aikana varsinaisten työtehtävieni määrittyessä tarkemmin.

Raportointiviikolla yksi, jota käsitellään tarkemmin luvussa 3.1, käynnistyi Robot Frameworkin harjoittelu käytännön kautta, jotta sain saavutettua tarvittavan osaamisen ja tietotason kyseisessä työkalussa mahdollisimman nopeasti. Kesätyösuhteen alkuvaiheessa tarvittavien pohjatietojen kerääminen oli jatkuvaa, joten kollegoideni apu sekä intra- että internetistä löytyvät lähteet olivat vielä tässä vaiheessa välttämättömiä.

Omaan kehitykseeni työelämässä ovat ennen kesätyösuhdettani Telialla vaikuttaneet erilaiset työkokemukset kesätyösuhteista puolen vuoden mittaiseen työharjoitteluun. Näiden kokemusten kautta olen mielestäni oppinut paljon erilaisia käytännön toimintatapoja ja tämä näkyy toiminnassani siten, että toimintani on motivoitunutta, varmaa, tarkkaa ja omaaloitteista. Tietenkin eri työpaikoilta löytyvät omat ”talon tavat”, mutta olen huomannut, että yleinen reipas ja vastuuntuntoinen työskentelytapa toimii missä tahansa. Kokemuksen kautta olen löytänyt myös vakiintuneen tavan viestittelyyn ja muuhun kommunikaatioon työpaikan eri sidosryhmien kanssa. Tämä opittu tapa on tähän mennessä toiminut hyvin työpaikasta ja sidosryhmistä riippumatta.

Käytännön työskentelytapojen ollessa minulle selkeät, mielestäni tällä hetkellä ainoa kehittämisen arvoinen asia omassa toiminnassani on ammatillisen osaamisen, eli tietotekniikan eri osa-alueisiin liittyvän tietotaidon, kehittäminen. Oma ammatillinen kehitykseni on vielä alkuvaiheessa Telian ollessa ensimmäinen oman alan töitä tarjoava kesätyöni. Aikaisempaa oman alan kokemusta olin saanut opintoihini liittyvästä työharjoittelusta Helsingin kaupungin Kaupunginkanslian tietotekniikka- ja viestintäosastolla. Kyseisessä työharjoittelussa pääsin muun muassa tekemään manuaalista testaustyötä sekä vastaamaan master-datan ylläpidosta sekä tiedon laadun parantamisesta ja säilyttämisestä. Tästä syystä kesätyöni Telialla olikin erinomainen mahdollisuus saavuttaa tämä tavoite ja kehittää omaa ammatillista oppimistani enteenpäin tulevia työsuhteita, -tehtäviä ja opintoja varten.

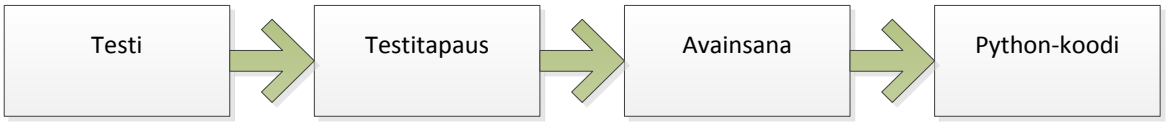
Mielestäni sekä yleinen että ammatillinen kehittyminen ja oppiminen ovat yhdet tärkeimmistä päämääristä työelämässä ja parhaiten ne tapahtuvat omasta kokemuksestani erilaisten onnistumisien, mutta etenkin myös virheiden kautta. Vaikka onnistuminen onkin epäonnistumista mieltä ja itsetuntoa ylentävämpi kokemus, virheitä tekemällä työntekijällä on hyvä mahdollisuus oppia näistä virheistä. Kyse ei tietenkään ole tahallisesta töiden laiminlyömisestä ja huolimattomuudesta vaan yrityksestä ja erheestä: yritys, epäonnistuminen, uusi yritys ja onnistuminen.

Vaikka teoriapohja on lähes aina edellytys erilaisten työtehtävien suorittamiseksi ja erilaisten järjestelmien, laitteiden ja sovelluksien käyttämiseksi, käytännön kautta oppiminen on yhtä tärkeä osa kehitysprosessia. Esimerkiksi ollessani töissä Helsingin kaupungin Palvelukeskuksessa kaupungin puhelinvaihteessa, tarvitsin ensin vankan teoriapohjan käytettävästä puhelunvälitysjärjestelmästä ja kaupungin organisaatiosta, jotta osaisin etsiä ja yhdistää saapuvat puhelut oikeille henkilöille. Työskennellessäni taitoni karttuivat ja näin järjestelmän toiminnallisuudet ja ominaisuudet jäivät sekä mieleen että lihasmuistiin. Kokemuksen ja muutaman väärin yhdistetyn puhelun kautta, kehittymistä tapahtui jatkuvasti tiedon ja taidon karttuessa.

2.1.1 Robot Framework ja PyCharm

Tämän kesätyösuhteeni aikana työtehtäviini kuului Telian eri järjestelmiä testaavien testiautomaatioiden ohjelmointi Robot Frameworkilla. Robot Framework on avainsanapohjainen, avoimen lähdekoodin geneerinen testiautomaatioviitekehys (Robot Framework Foundation, 2017b). Testien automatisointi Robot Frameworkilla perustuu Python-koodilla ohjelmoitujen avainsanojen käyttöön, joiden avulla pystytään määrittämään erilaisia toimintoja, joita Robotti automaattisesti suorittaa.

Tässä tapauksessa web-käyttöliittymää testattaessa, Robotti pystyy tähän tarkoitukseen luotujen avainsanojen kautta muun muassa avaamaan internet-selaimen ja klikkaamaan haluttua web-elementtiä. Halutut avainsanat saadaan käyttöön importoimalla testitiedostoon eri Robot Frameworkin kirjastoja, erikseen määriteltyjä resurssitiedostoja tai luomalla avainsanat suoraan avainsanataulukkoon (Robot Framework Foundation, 2017c). Käytetyt avainsanat muodostavat erilaisia suoritettavia kokonaisuuksia, testitapauksia, ja testitapaukset taas muodostavat yhdessä ajettavan testin. Tämä Robot Framework-testien rakenne on kuvattuna alla olevassa kuvassa 1, ” Robot Framework-testin rakenne”.



Kuva 1 Robot Framework-testin rakenne

Robot Frameworkilla automatisoitujen testien ohjelmointi tapahtui avoimen lähdekoodin integroidussa ohjelmointiympäristössä, PyCharmissa (dvs1, 2017), joka on tarkoitettu sovelluskehitykseen erityisesti Pythonilla. Alkuperäisestä käyttötarkoituksestaan huolimatta PyCharmiin on mahdollista ladata liitännäinen PyCharmin kehittäjän JetBrainsin sivuilta, joka sisältää tuen Robot Frameworkille (JetBrains, 2017b). Näin ollen PyCharm soveltuu loistavasti myös Robot Frameworkin ohjelmoinnin alustaksi.

2.1.2 Työtehtävät

Vaikka tiesin työsuhteeni alkaessa, että työnkuvaani kuuluisivat eri järjestelmien testaaminen niiden web-käyttöliittymien kautta, vasta kolmannella raportointiviikolla minulle selvisi minkä järjestelmien automaattisesta testaamisesta tulisin vastaamaan. Työnkuvaani kuului Telian verkkokaupan, erään myyntijärjestelmän ja asiakkaan omien sivujen testaaminen kyseisten järjestelmien web-käyttöliittymän kautta. Kyseessä olivat järjestelmien testi-instanssit, jotka toimivat testiympäristössä ja tämän ansiosta pystyin työstämään testejä ilman pelkoa siitä, että esimerkiksi automatisoimani tilaukset tulisivat näkyviin tuotannon puolelle.

Verkkokaupasta sekä uudet että vanhat Telian asiakkaat pystyvät ostamaan yrityksen tarjoamia tuotteita ja palveluita. Myyntijärjestelmä on järjestelmä, jonka kautta Telian työntekijä, esimerkiksi asiakaspalvelija tai myyjä, voi tilata myyntitilanteessa asiakkaalle tuotteita ja palveluita. Asiakkaan omat sivut ovat järjestelmä, jonka web-käyttöliittymän kautta asiakas pystyy muun muassa hallinnoimaan omia tilauksiaan ja niihin liittyviä asiakastietoja. Näille kolmelle järjestelmälle minun oli tehtävänä luoda web-käyttöliittymän kautta suoritettavia, automaattisia testejä, joita olisi myöhemmin mahdollista käyttää pohjana regressiotestauksessa kyseisille järjestelmille. Työtehtävänäni oli ohjelmoida Robot Frameworkilla seuraavat prosessit kyseisissä järjestelmissä:

Verkkokauppa

- Puhelimen tilaus kertamaksulla
- Puhelimen tilaus Telia rahoituksella/ kuukausimaksulla
- Laajakaistaliittymän tilaus
- Puhelinliittymän tilaus

Myyntijärjestelmä

- Laajakaistayhteyden ja Spotify-palvelun tilaus kaapelitalouteen
- Laajakaistayhteyden ja tietoturvapalvelun tilaus kuitutalouteen
- VPN-palvelun tilaus
- TV-palvelun tilaus
- Laitevakuutuksen tilaus

Asiakkaan omat sivut

- Laajakaistayhteyden laskutustietojen muuttaminen
- Mobiililiittymän laskutustietojen muuttaminen
- Kanavapakettien tilaaminen TV-palveluun
- Laajakaistayhteyden nopeuden nostaminen

Konkreettisesti työtehtävissäni ensin tutustuin testattaviin järjestelmiin ja perehdyin automatisoitaviin toimintoihin. Oman työni ja oppimiseni tehostamiseksi dokumentoin prosessit vaihe vaiheelta ylös, jotta näin selkeästi mikä olisi automatisoitavan prosessin seuraava vaihe sekä suoritettavien toimintojen halutut lopputulokset. Automatisoitavan prosessin ollessa minulle itselleni selvä, aloitin testin ohjelmoimisen Robot Frameworkilla PyCharm-kehitysympäristössä, jonne loin omat robottitiedostot jokaiselle testattavien järjestelmien prosesseille. Tämän jälkeen lähdin omaa dokumentaatiotani seuraten ohjelmoimaan varsinaisia testejä.

Suoriutuakseni työtehtävistäni, minun oli osattava ohjelmoida Robot Frameworkia niin, että pystyin ohjelmoimaan tarvittavia automatisoituja testejä. Robot Frameworkin osaamisen lisäksi työtehtävät vaativat tarpeellista tietämystä testattavista järjestelmistä, automatisoitavista prosesseista ja yleisellä tasolla siitä, miten tieto kulkee testiympäristössä testattavien järjestelmien testi-instanssien välillä.

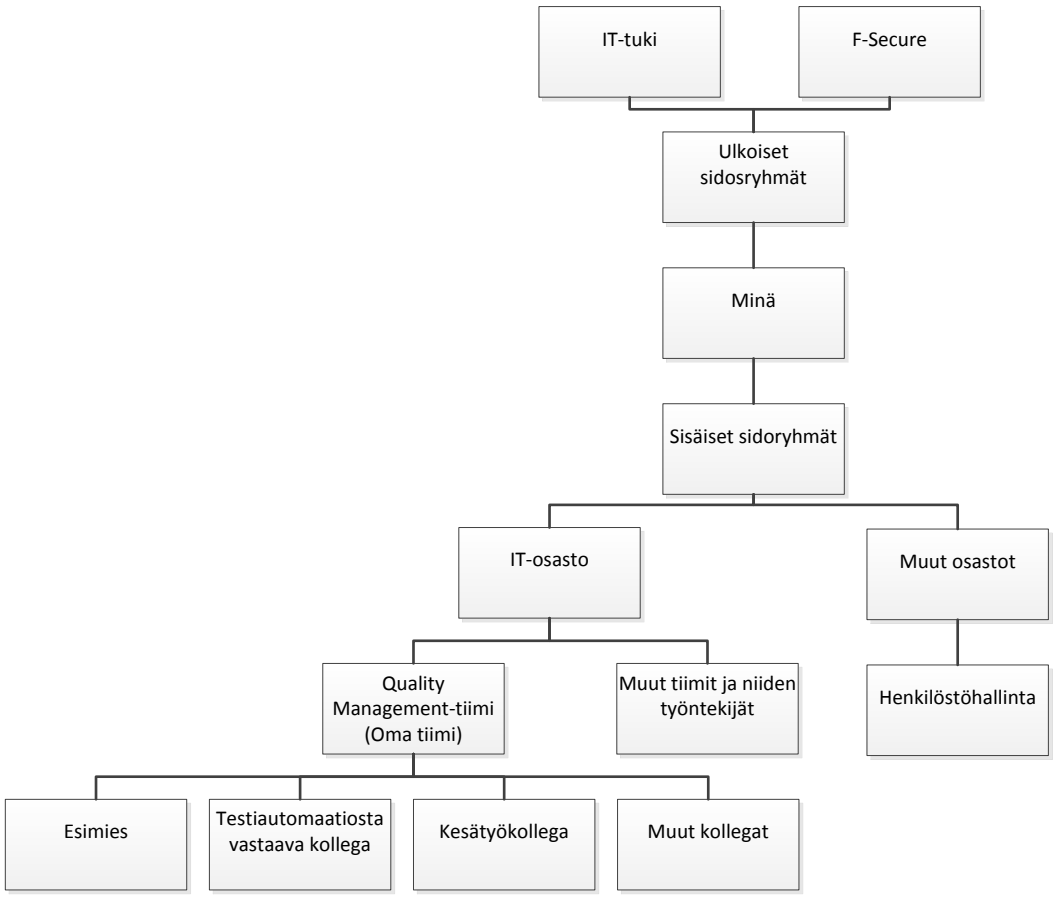
Työsuhteeni alussa oletin, että tulisin tarvitsemaan myös Python-ohjelmointikielen ohjelmointiin liittyviä taitoja. Tästä syystä tein opinnäytetyön aloitusviikolla muutaman päivä ajan erilaisia Python-ohjelmoinnin perusteisiin liittyviä harjoituksia ymmärtääkseni hieman tarkemmin Pythonin logiikkaa ja sitä, miten Robot Frameworkin avainsanat on ohjelmoitu. Pythonin harjoittelussa hyödynsin Code Academy-alustaa, joka on verkko-opetusta tarjoava yritys (Code Academy, 2017). Ensimmäisten viikkojen aikana kävi kuitenkin ilmi, että en tulisi tarvitsemaan Python-ohjelmointitaitoja tämän kesätyösuhteeni aikana.

2.2 Sidosryhmät työpaikalla

Telian sidosryhmät jakautuvat kahteen eri ryhmään, yrityksen sisäisiin ja ulkoisiin. Sisäisiin sidosryhmiin kuuluvat kollegat ja esimiehet niin omasta kuin muistakin tiimeistä, muut eri vastuualueiden osastot sekä muut organisaation osat kuten johtoporras. Yrityksen ulkopuolisiin sidosryhmiin kuuluvat yksityiset ja yritysasiakkaat, yhteistyökumppanit ja muut ulkoistetut toimijat kuten, Telian järjestelmien toimittamisesta ja ylläpidosta vastaavat yritykset sekä muut teleoperaattoriyritykset.

Vaikka Telia on hyvin laaja ja moniin maihin levittäytynyt yritys, tiettyihin sekä sisäisiin että ulkoisiin sidosryhmiin suuntautuva vuorovaikutus voi olla yksipuoleista, epäsuoraa ja jopa harvinaista. Tässä opinnäytetyössä tarkastellaan ja kuvataan vain sellaisia sidosryhmiä, joihin olen kesätyösuhteeni aikana ollut itse molemminpuolisessa vuorovaikutuksessa. Nämä sekä sisäiset, että ulkoiset sidosryhmät ovat kuvattuna kuvassa 2 ”Työpaikan sidosryhmät”. Tällaisiin sisäisiin sidosryhmiin kuuluvat oma tiimi ja esimies, jotka ovat myös työni kannalta keskeisimmät sidosryhmäni saadessani heiltä suoraan apua, palautetta, uusia työtehtäviä, perehdytysmateriaaleja ja koulutusta niin varsinaisista työtehtävistä ja niissä käytettävistä työkaluista kuin myös koko organisaatiosta, sen rakenteesta ja toiminnasta. Vastineeksi he saavat minulta esimerkiksi työtuloksia tai töihin liittyviä kysymyksiä ja palautetta. Muista sisäisistä sidosryhmistä olen vastavuoroisesti yhteydessä muihin tarvittaviin tiimeihin ja osastoihin sekä henkilöstöhallintoon.

Ulkoisista sidosryhmistä olin molemminpuolisessa vuorovaikutuksessa ainoastaan ulkoistettuun it-tukeen ja F-Secure-yritykseen. IT-tuelta sain apua ongelmiin ja kysymyksiin liittyen testaamiini ja käyttämiini järjestelmiin. Vastineeksi ilmoitin heille vastaan tulleista vikatilanteista ja testasin, toimivatko heidän tekemänsä korjaukset. F-Secureen, jouduin ottamaan yhteyttä liittyen myyntijärjestelmän kautta tehtyihin VPN- ja tietoturvapalvelutilauksiin, jotka olivat jääneet jumiin ja hankaloittivat testien automatisointia ja uudelleen ajamista. Kuten it-tuen kanssa, pystyin vastineeksi ilmoittamaan F-Securelle vikatilanteiden etenemisestä ja mahdollisien korjauksien toimivuudesta.



Kuva 2 Työpaikan sidosryhmät

2.3 Vuorovaikutustaidot työpaikalla

Työskentelyyn Telialla liittyi erilaisia vuorovaikutustilanteita kollegoideni ja esimieheni kanssa. Yleisin vuorovaikutustilanne varsinkin testiautomaatiosta vastaavan kollegani kanssa oli koulutustilanne tai neuvojen kysyminen Robot Frameworkiin ja muihin työssä käyttämiini työkaluihin sekä työtehtäviin liittyen. Samalla tavalla sain apua muiltakin kollegoiltani esimerkiksi testaamiini ja käyttämiini järjestelmiin liittyviin kysymyksiin.

Olin varsinkin oman tiimini kollegoihin yhteydessä pääasiassa kasvotusten heidän istuessa samassa kerroksessa ja tilassa kanssani. Jos he eivät olleet paikalla, he olivat varattuja tai heidän toimipisteensä sijaitsi muualla kuin Helsingissä, olin heihin yhteydessä pikaviestintäsovelluksen Skypen tai sähköpostin kautta. Sama toimintaperiaate päti myös esimieheni kohdalla. Hänen kanssaan vuorovaikutustilanteet liittyivät yleensä koulutukseen, jotain organisaation toimintaan ja käytäntöihin liittyvää kysymystä esimerkiksi kesätyöntekijöiden lomapäiviin ja työterveyshoitoon liittyen.

Muihin kuin oman tiimini jäseniin olin pääsääntöisesti yhteydessä pikaviestintäsovelluksella tai sähköpostitse liittyen erilaisiin ongelmatilanteisiin ja kysymyksiin. Teknisiin ongelmiin

liittyen olin niistä vastaaviin henkilöihin epäsuorasti yhteydessä myös tehtävänhallintaohjelmisto Jiran kautta ilmoittaessani heille löytämistäni bugeista ja vastaan tulleista ongelmista. Puhelimitse en ollut muuten yhteydessä kuin asiakaspalveluun selvittäessäni työpuhelimeeni liittyvää ongelmatilannetta.

Erilaiset vuorovaikutustilanteet eivät tuottaneet minulle haasteita, koska olin aikaisempien työkokemuksieni kautta sisäistänyt hyväksi näkemäni toimintatavat ollessani erilaisten ihmisten kanssa vuorovaikutuksessa. Tästä johtuen minun oli helppo olla ja toimia erilaisissa vuorovaikutustilanteissa erilaisten ihmisten kanssa lähestyivät he minua, tai minä heitä.

Telian ollessa moneen maahan levittäytynyt organisaatio, jossa työskentelee ihmisiä monesta eri maasta ja kulttuurista, ainoana uutena asiana vuorovaikutukseen liittyen minulle oli kommunikointi englanniksi tiettyjen sidosryhmien kanssa. Tällainen sidosryhmä oli esimerkiksi Suomen ulkopuolelle ulkoistettu it-tuki. En ollut aikaisemmissa työsuhteissani ollut näin aktiivisesti kommunikoinut minkään sidosryhmän kanssa pelkästään englanniksi. Olin Helsingin kaupungin puhelinvaihteessa joutunut päässyt palvelemaan asiakkaita myös englanniksi, mutta ne olivat harvoja ja yksittäisiä tapauksia.

3 Päiväkirjaraportointi

3.1 Seurantaviikko 1

Maanantai 12.6.2017

Ensimmäisen raportointipäivän tavoitteena oli oppia lisää Robot Frameworkista ja sen käytöstä, sen ollessa tärkein työkalu tulevissa työtehtävissäni. Tarkoituksenani oli perehtyä tarkemmin aiheeseen Robot Frameworkin kotisivuilta löytyvän käyttöohjeen, Robot Framework User Guiden (Robot Framework Foundation, 2017c) avulla. Teoreettinen ja laaja dokumentaatio kattaa käyttöohjeen mukaisesti Robot Frameworkin eri osa-alueet aina työkalun esittelystä ja asentamisesta, sen käyttötarkoitukseen ja sen eri komponenttien, kuten avainsanakirjastojen, toiminnallisuuksiin.

Päivälle asettamani tavoitteet toteutuivat suunniteltua paremmin, koska pelkän teoriaan tutustumisen sijasta päätin perehtyä Robottiin myös itsenäisen käytännön harjoittelun kautta seuraamalla Robot Frameworkin kotisivuilta löytyviä, videoituja esimerkkejä. Näissä esimerkkivideoissa kirjoitettiin testejä, joissa Robotti oli ohjelmoitu avaamaan internetselain, etsimään ennalta määritelty hakusana Googlesta, tarkistamaan löytyykö kyseistä hakusanaa vastaava internetsivu hakutuloksista ja avaamaan etsityn sivun (Robot Framework Foundation, 2017f).

Toistin katsomani esimerkkitestit PyCharmilla muuttamalla demossa käytettyjen muuttujien arvoja, kuten hakusanaa, käytettävää selainta ja etsittävää internetsivustoa. Omassa testissäni Robotti esimerkiksi avaa Chromen sijasta Firefox-selaimen ja etsii hakutuloksista ”Levykauppa Äx”-hakusanaa vastaavaa internetsivustoa. Oman oppimisen tehostamiseksi lisäsin testiin lisää vaiheita, joissa avattuaan Levykauppa Äx:n sivun, Robotti syöttää halutun albumin nimen sivulta löytyvään hakukenttään ja klikkaa hakupainiketta.

Vaikka toistamani testi oli hyvin yksinkertainen, se vei silti vasta aloittelevalta Robot Frameworkin ohjelmoijalta, kuten minulta, useamman tunnin. Olin kuitenkin tyytyväinen saavuttamaani tulokseen, koska käytännön harjoittelun kautta sain selkeämmän kuvan siitä, miten Robot Framework käytännössä toimii. Testiä ohjelmoidessani sain samalla myös tarvittaessa suuntaa-antavia neuvoja testiautomaatiosta vastaavalta kollegaltani, kuka vastaa minun ja kesätyökollegani Robot Framework koulutuksesta.

Tämä päivän ohjelmaan kuului Robot Frameworkiin perehtymisen ohella myös nopea katsaus Jenkins-nimiseen työkaluun. Jenkins on jatkuvaan integraatioon tarkoitettu applikaatio, jonka avulla käyttäjät pystyvät ajamaan omia järjestelmiään testaavia automatisoituja testejä (Shatzer, 2016). Kyseisellä työkalulla pystytään ajamaan halutut Robot Framework testit valituissa virtuaaliympäristöissä testien suorittamisen automatisoimiseksi ja näin ollen testaamaan järjestelmiä esimerkiksi aina silloin, kun niihin tehdään muutoksia. Jenkinsiin ja sen käyttöön ei ole kuitenkaan tarkoitus perehtyä tarkemmin ainakaan näin aluksi sen selkeän käytettävyyden ja käyttötarkoituksen takia ja siksi, että työskentelyn pääpaino tulee olemaan Robot Frameworkin käyttämisessä ja hallinnassa.

Tiistai 13.3.2017

Tämän päivän tavoitteena oli jatkaa siitä, mihin jäin eilen Robotilla tehdyn testiautomaatioharjoituksen kanssa ja sain päivän tavoitteeni saavutettua. Valittuaan halutun levyn saaduista hakutuloksista nimen ja tuotteen id-numeron kautta, Robotti tarkistaa onko kyseistä tuotetta saatavilla jossain Levykauppa Äxän toimipisteistä ja lisää saatavilla olevan tuotteen ostokoriin. Päivän toisena tavoitteena oli aloittaa uuden testiautomaatioharjoituksen ohjelmoiminen, jossa Robot Framework avaa Telian verkkokaupan sivun, valitsee ennalta määritellyn matkapuhelimen ja siirtää sen ostoskoriin. Tämän jälkeen Robotti täyttää vielä tilaajan tiedot ja viimeistelee tilauksen sen lähettämistä vaille valmiiksi.

Uutta testiautomaatioharjoitusta ohjelmoidessamme törmäsimme kesätyökollegani kanssa yhteensopivuusongelmaan testeissä käyttämämme Firefox-selaimen ja Robot Frameworkin kanssa. Saimme ongelman iltapäivän aikana ratkaistua testiautomaatiosta vastaavan kollegamme avulla asentamalla selaimesta 32-bittisen version, 64-bittisen version sijasta. Tämän takia emme päässeet kesätyöntekijäkollegani kanssa aloittamaan uuden, Telian verkkokauppaa testaavan testin ohjelmointia kuin vasta iltapäivällä. Uusi testi tulee kuitenkin olemaan lähes samanlainen kuin tekemäni Levykauppa Äx-harjoitus, joten testin ohjelmointiin ei pitäisi mennä kauaa vaikka testattavat sivut ovatkin täysin erilaiset.

Keskiviikko 14.6

Kolmannen raportointipäivän tavoitteena oli eilen aloittamani uuden, Telian verkkokauppaa testaavan testiautomaatioharjoituksen jatkamista vaiheeseen, jossa uuden tuotteen tilaus on tilauksen vahvistamista vaille valmis. Vaikka tavoite oli melko korkea yhdelle päivälle ja omaan Robot Framework osaamiseeni nähden, tavoite oli haastavuutensa takia erinomainen oppimisprosessi, tuli tavoite saavutettua tai ei. Uuden testin saaminen päivässä loppuun osoittautui odotetusti hieman liian korkeaksi tavoitteeksi yhdelle päivälle:

ongelmaksi nousi tilattavan tuotteen tuotesivulta löytyvät valintapainikkeet eri maksutavoille ja laitevakuutukselle, joita Robotti ei jostain syystä saanut aluksi aktivoitua.

En kuitenkaan keskittynyt kokopäiväisesti uuden testiautomaation viimeistelyyn, vaan koska minulla oli aikaa, lähdin kokeilemaan myös eri tapoja, joilla Robot Framework pystyy paikantamaan halutun elementin html-koodista. Hyväksi tavaksi osoittautui elementtien yksilöivien nimien ja id-tunnusten käyttäminen elementtien XPath-polkuja hyödynnettäessä. XPathilla voidaan navigoida XML-dokumentissa hyödyntämällä sen polkumaista syntaksia (W3Schools, 2017c) ja sen avulla voidaan etsiä myös html-koodista haluttuja elementtejä.

Tämän kokeilun tarkoituksena oli löytää vaihtoehtoisia etenemistapoja nopeimman ja helpoimman vaihtoehdon sijasta, koska hankaluuksia aiheutti eritoten useamman attribuutin hakeminen samanaikaisesti erilaisten html-elementin sisältä. Toimivimmaksi tavaksi osoittautui paikantaa halutut elementit koodista yksi kerrallaan niiden XPath-polkuja ja eri Robotin avainsanoja hyödyntämällä. Näin koodi pysyi selkeänä ja vältyttiin myös turhan pitkiltä ja monimutkaisilta yksittäisiltä lauseilta.

Vaikka en päässytkään kesätyökollegani kanssa päivälle asetettuun tavoitteeseen, saimme toiston, kokeilun ja virheiden kautta opittua hyvin, miten tarvittavia html-elementtejä kannattaa Robotilla hakea. Samalla sain myös vakiinnutettua tapaa, jolla kirjoitan Robot Frameworkin koodia niin, että sen rakennetta on entistä helpompi lukea myös sellaisen ulkopuolisen käyttäjän, jolle Robot Framework ei ole entuudestaan tuttu. Pyrin esimerkiksi nimeämään Robotin testitapaukset niin, että niiden nimestä saa käsityksen siitä, mitä ne tulevat tekemään. Kuten Robot Frameworkin GitHub-sivustolla kirjoitetaankin (pekka-klarck, TechieWidget & charleswhchan, 2017), testitapausten nimien tulisi olla mahdollisimman kuvaavia ja sopivan mittaisia. Tällainen koodin oikeaoppinen muotoilu helpotti myös omaa työtäni huomattavasti ja auttoi ymmärtämään Robotin toimintaa entistä paremmin.

Torstai 15.6

Torstain agenda koostui taas pääasiallisesti Robot Frameworkin itseopiskelusta tiimini testiautomaation versionhallinnasta ja internetistä löytyvien materiaalien avulla. Päivän tavoitteena oli saada jatkettua tiistaina aloittamaani Telian verkkokaupan testiautomaatioharjoitusta eteenpäin. Tämä tavoite tuli hyvin saavutettua ohjelman edettyä siihen pisteeseen, jossa Robotti lisää tilattavan tuotteen ostoskoriin avattuaan verkkosivun ja etsittyään kyseisen laitteen.

Päivän aikana tapahtui selkeää kehitystä Robot Frameworkin ohjelmointiin liittyen: yrityksen ja erheen kautta sain harjoiteltua miten erilaisten painike-elementtien löytäminen ja käyttäminen luonnistuvat Robotin kautta. Päivän aikana huomasin myös, miten ajettavan Robot Framework testin suoritusnopeus voi vaikuttaa testin onnistumiseen. Esimerkiksi jos testin suoritusnopeus on liian suuri, web-sivu ei välttämättä ehdi tarpeeksi nopeasti ladata tarvittavia elementtejä, jolloin Robotti ei pysty paikantamaan kyseisiä web-elementtejä ja testi epäonnistuu.

Perjantain tavoitteeksi jäi viedä työstämäni harjoittelutesti loppuun. Tavoite on melko suuri yhdelle päivälle, mutta silti saavutettavissa, koska kaikkia testin viimeistelyyn vaadittavia osa-alueita, kuten tekstin syöttämistä tekstikenttään, erilisten painikkeiden painamista, sivulta toiselle liikkumista ja sivun sisällä navigointia, on tullut tämän viikon aikana jo harjoiteltua. Vastaan voi kuitenkin aina tulla erilaisia ongelmatilanteita, jotka aiheuttavat viivästymisiä ja estävät asetettuiin tavoitteisiin pääsyn.

Perjantai 16.6

Tämän päivän tavoitteena oli saada tiistaina aloittamani Telian verkkokaupan testiautomaatioharjoitus valmiiksi. Vaikka sain kesätyökollegani kanssa työstettyä kyseistä testiä päivän aikana, päivän tavoite ei kuitenkaan tullut täysin saavutettua. Jotta asiakkaan suoramaksulla tehtävä matkapuhelintilaus saataisiin viimeisteltyä, verkkokaupan käyttäjän tulee ostokorin sisällön varmistamisen jälkeen tunnistautua palveluun joko olemassa olevilla asiakastunnuksilla, mobiilivarmenteella tai pankkitunnuksilla. Emme olleet saaneet vielä testiverkkopankkitunnuksia, koska kyseessä oli oma-aloitteista ja itsenäistä harjoittelua tuotantoympäristössä.

Koska tilaajan tietojen syöttäminen ei ollut mahdollista pakollisen tunnistautumisvaiheen takia, päätimme kollegani kanssa lisätä testiin muutama uusi vaihe harjoittelun hyödyn maksimoimiseksi. Kun halututtu tuote on lisätty ostoskoriin, käyttäjän on mahdollista valita ostettavalle laitteelle jokin lisälaite, kuten kuulokkeet tai kannettava lisävirtalähde. Tämä vaihe oli helppo automatisoida, koska lisälaiteen valitsemisessa pystyi hyödyntämään tilausprosessin aikaisemmissa vaiheissa käytettyjä Robot Frameworkin avainsanoja. Samalla kun Robotti valitsee halutun lisälaitteen, se valitsee myös kahdesta eri alasvetovalikosta halutun maksuvaihtoehdon ja lisälaitteen värin. Tämän jälkeen Robotti vielä syöttää alennuskoodi-kenttään keksityn alennuskoodin ja tarkistaa, että sivu antaa virheilmoituksen väärästä koodista.

Kun saimme testin viimeisteltyä tunnistautumista lukuun ottamatta, lähdin hiomaan koodia niin, että muuttujien, kuten laitteen ja tuotesivujen, arvot määritellään testitapausten ja avainsanojen ulkopuolella testitasolla. Näin muuttujien arvojen muuttaminen ja saman testin ajaminen toisilla arvoilla onnistuisi entistä helpommin. Tämän lisäksi ohjelmoimme kollegani kanssa testitapauksen, joka sulkee verkkokauppatestin aikakana mahdollisesti avautuvan pop-up-ikkunan, joka johti testin epäonnistumiseen peittäessään tietyt web-elementit. Testitapauksessa Robotti tarkisti, että oliko sivulle ilmestynyt pop-up-ikkuna ja jos oli, niin Robotti sulki sen.

Kyseistä tilannetta silmällä pitäen, testiautomaatiosta vastaavan kollegani mukaan Robot Frameworkilla ei kuitenkaan ole mahdollista suorittaa jatkuvaa web-sivun monitorointia muiden testitapausten yläpuolelle. Kollegani mukaan Robotti voi ajaa testitapaukset ja avainsanat vain määritellyissä kohdissa testiä, joten näin ollen pop-up-ikkunan esillä olon tarkistus oli asetettava siihen kohtaan testiä, jossa se yleensä ilmestyi näkyviin. Tämän tarkistuksen ohjelmoimisessa hyödynsimme yhteisestä versionhallinnasta löytyvää myyntijärjestelmän testiautomaatioesimerkkiä, johon oli tehty samankaltaisia tarkistuksia. Näistä esimerkeistä oli suuri hyöty omaa testitapausta tehdessä ja näin ollen ne toimivat hyvänä oppimateriaalina.

3.1.1 Viikkoanalyysi 1

Ensimmäisen raportointiviikon aikana oppimiseni painottui Robot Frameworkin opetteluun niin itsenäisesti kuin ohjatusti. Itsenäisessä opiskelussa materiaaleina toimivat Robot Frameworkista löytyvät sähköiset materiaalit, kuten Robot Framework User Guide (Robot Framework Foundation, 2017c). Verkkosivujen testaamiseen liittyvän käytännön harjoittelun apuna toimivat erinomaisesti erilaiset valmiit esimerkkitestiautomaatiot, jotka saimme kesätyökollegani kanssa käyttöömme ja tutkittavaksi tiimimme testiautomaation versionhallinnasta jo viime viikon loppupuolella.

Tämän viikon aikana varsinkin Robot Frameworkiin liittyvä osaamiseni kehittyi nopeasti. Ennen työsuhteeni alkua, noin kaksi ja puoli viikkoa sitten, en ollut aikaisemmin edes kuullut Robot Frameworkista ja tietoperustani testiautomaatioon liittyen oli myös hyvin vähäistä. Kanerin, Bachin ja Pettichordin (2002, s.115) mukaan avainsanapohjainen testiautomaatio onkin hyvä ei-ohjelmoijille luoda omia automatisoituja testejä. Huomasin tämän myös itsekin vaikka ohjelmointi, varsinkaan Pythonin ohjelmointi, ei ole yksi omista vahvuuksistani, pääsin automaattisten testien kirjoittamisessa melko nopeasti alkuun Robot Frameworkin avulla.

Ensimmäinen työviikkoni, eli opinnäytetyön aloitusviikko, josta ei erillistä päiväkohtaista raporttia ole, meni pääsääntöisesti Robottiin liittyvään materiaaliin perehtymisessä muihin välineisiin, järjestelmiin ja organisaation toimintaan liittyvän perehdytyksen ohella. Ensimmäisen raportointiviikon alkaessa minulla alkoi olla jo käsitystä siitä, mihin Robot Framework on tarkoitettu, mutta sen varsinaisesta toiminnasta minulla ei ollut vielä selkeää kuvaa pelkkään teoriaan tutustumisen kautta. Vasta aloitettuani tällä viikolla itsenäisesti harjoittelemaan Robotin käyttöä eri käytännön esimerkkien kautta, sain vihdoin paremman käsityksen sen toiminnasta.

Viikon alussa tuntui hieman turhauttavalta, että vaikka työsuhdetta oli kestänyt jo melkein kaksi ja puoli viikkoa, varsinaiset työtehtäväni eivät ole vielä määrittyneet. Ymmärsin kuitenkin sen, että ennen kuin voin aloittaa testiautomaatioiden tekemisen varsinaisena työtehtäväni, minun on ensin opittava käyttämään työtehtäviin tarvittavia työkaluja. Mielestäni olisi kuitenkin hyvä tietää jo etukäteen minkä järjestelmien kanssa tulen työskentelemää, jotta voisin aloittaa niihin perehtymisen ohjattujen koulutustapahtumien välissä.

Tämän viikon jälkeen tuntui myös siltä, että julkisella sektorilla perehdyttämisen pääpaino oli enemmän ohjatussa perehdytyksessä vaikka tietysti siellä painotetaan paljon myös oma-aloitteisuuteen ja kykyyn selvittää asioita itsenäisesti. Kahden ja puolen viikon aikana olen huomannut, että vaikka Telialla on myös ohjattua perehdytystä käytettäviin järjestelmiin ja työkaluihin liittyen, oppimisen pääpaino on selkeästi itsenäisellä opiskelulla. Tähän viittasi se, että ohjattujen perehdytyssessioiden välissä oli melko paljon tyhjää tilaa, jolloin uutena työntekijänä minun oli aktiivisesti syvennyttävä itse läpikäytyihin materiaaleihin ja aiheisiin itse.

Välit perehdytyssessioiden välissä saattoivat tietysti myös johtua Telian suurista organisaatiomuutoksista Telian muuttuessa Sonerasta Teliaksi, jonka takia esimiehillä ja kollegoilla oli varmasti muutenkin kädet täynnä töitä. Minua ja kesätyökollegaani ohjattiin perehdytyssessioiden jälkeen perehtymään itsenäisesti läpikäytyihin materiaaleihin tarkemmin. Mutta esimerkiksi harjoitellessamme Robot Frameworkin ohjelmoimista käytännössä Telian verkkokaupan web-käyttöliittymää testaamalla, kukaan ei tullut erikseen sanomaan, että meidän tulisi tehdä käytännön harjoittelua erilaisten testiautomaatioesimerkkien kautta. Käytännön harjoittelu lähti meistä itsestämme liikkeelle, koska näimme, että näin saisimme opiskelusta ja saatavilla olevista materiaaleista kaiken mahdollisen irti.

Olen viikon mittaan kokeillut ja hyödyntänyt eri lähteistä vastaan tulleita toimintamalleja Robot Frameworkin ohjelmoimiseen liittyen. Huomasin esimerkiksi, että argumenttien käyttö Robotin koodissa oli hyödyllinen toimintatapa, joka helpotti ja selkeytti omaa ohjel-

moimista. Argumenttien avulla pystyin liikuttelemaan eri muuttujia vaivatta testin sisällä eri testitapauksissa. Koska muuttujat on määritelty testitapauksen ulkopuolella eikä kovakoodattu testitapauksien sisälle, niiden arvoja pystyy myös helposti muuttamaan yhdestä sijainnista.

Huomasin kuitenkin pian, että argumenttien käyttö ei ollut välttämätöntä. Eri muuttujien liikuttelu nopeasti ja vaivatta läpi testin ja testitapauksien oli lähes yhtä helppoa, ellei jopa helpompaa ilman argumentteja. Tämä tapahtui määrittelemällä tarvittavat muuttujat jo testitasolla tai asettamalla testitapausten sisällä määriteltävät muuttujat testikohtaisiksi muuttujiksi testitapauksen sisällä. Kokeilemalla erilaisia toimintamalleja ja – tapoja varsinkin vasta opetellessani uuden viitekehyksen käyttöä, se on ollut hyvä mahdollisuus löytää itselleen se sopivin ja toimivin malli, jota voi hyödyntää jatkossa.

Kuten aikaisemmin jo mainitsinkin, ensimmäisen raportointiviikon pääpaino oli Robottiin tutustuminen web-sivujen testaamisen kautta. Kuluneen viikon aikana jouduin selvittämään miten Robotti esimerkiksi löytää tarvittavat elementit html-koodista ja vuoro vaikuttaa näiden kyseisten elementtien, kuten erilaisten painikkeiden ja tekstikenttien, kanssa.

Elementtien paikantaminen manuaalisesti internetsivulta tapahtui etsimällä ne halutulta sivulta manuaalisesti erilaisten Firefox-selaimen tarkasteluun tarkoitettujen työkalujen, kuten Firebugin avulla. Firebug on Firefox-selaimen liitännäinen, jonka avulla käyttäjä pystyy erilaisten kehitystyökalujen avulla tutkimaan esimerkiksi halutun sivun html-koodia (Mozilla Foundation, 2017a). Tämän jälkeen kyseisillä työkalulla katsottiin, mikä olisi elementin selkein yksilöivä attribuutti, jolla haluttua elementtiä voidaan Robotilla etsiä ja tunnistaa. Esimerkiksi elementin id-tunnus tai nimi olivat hyviä vaihtoehtoja. Kun tarvittava attribuutti oli löytynyt, Robotin kirjastoista etsittiin ne avainsanat, jotka sopivat tilanteeseen ja käyttötarkoitukseen parhaiten. Avainsanan arvoksi annetaan etsittävän elementin XPath-polku, jolla Robotti paikantaa halutun elementin html-koodista.

Viikon aikana vastaani tuli useita eri ongelmia liittyen Robot Frameworkiin, mutta nämä ongelmat selvisivät kokeilemalla, kysymällä kollegoilta neuvoa, pohtimalla niitä yhdessä kesätyökollegan kanssa, turvautumalla annettuun oppimateriaaliin ja etsimällä vastauksia internetistä. Pääsääntöisesti ongelmanratkaisu tapahtui itse selvittämällä ja kokeilemalla niin pitkälle kuin vain oli mahdollista, ennen avun kysymistä. Näin sain itse opittua ongelmatilanteista hyvin paljon lähtemällä kokeilemaan eri vastausvaihtoehtoja ja pohtimaan, mistä saatu virhe johtui. Jos tuntui, että ongelma ei syystä tai toisesta ratkennut, päädyin kysymään apua, jotta minulta ei menisi liikaa aikaa ongelmatilanteen selvittämiseen.

3.2 Seurantaviikko 2

Maanantai 19.6.2017

Tälle päivälle en ollut varsinaisesti asettanut muita tavoitteita kuin viime viikolla aikaiseksi saamani koodin siistiminen ja hiominen. Web-sivujen testiautomaatiota koskevan oppimisen tehostamiseksi päätimme kesätyökollegani kanssa ohjelmoida uuden testitapauksen, jossa Robotti hakee ennalta määritellyn laitteen suoraan Telia verkkokaupasta navigointipalkista löytyvän hakukentän kautta.

Kyseisessä testitapauksessa Robotti syöttää haettavan laitteen nimen hakukenttään ja tarkistaa, että kyseinen laite löytyy hakukoneen automaattisesti näkyville tulevista vaihtoehdoista. Jos etsitty laite löytyy hakuvaihtoehdoista, Robotti avaa valitun laitteen tuotesivun. Tuotteen haun jälkeen tilausprosessia oli helppo jatkaa vain lisäämällä uuden testitapauksen jälkeen viime viikolla tehtyjä valmiita testitapauksia, kuten tuotteen lisääminen ostoskoriin sekä maksuvaihtoehtojen ja lisälaitteiden valitseminen, prosessin loppuun viemiseksi. Saimme testin toimimaan myös käyttämällä eri muuttujien arvoja ja näin hakemalla Telian verkkokaupasta eri tuotteita.

Kyseisessä hakutestissä saimme samalla myös testattua käytännössä viime viikolla ohjelmoimamme testitapauksen, joka sulkee Telia verkkosivulle aukeavan pop-up-ikkunan. Kyseinen ikkuna saattoi aueta itsestään missä vaiheessa tilausta tahansa, jos käyttäjä jäi esimerkiksi sivustolla liian pitkäksi aikaa paikoilleen. Asetimme pop-up-ikkunan sulkevan testitapauksen siihen kohtaan tilausprosessia, jossa pop-up-ikkuna yleensä aukesi ja testi toimi niin kuin pitikin. Tietyssä tilausprosessin vaiheessa Robotti tarkisti löytyikö sivulta auennut pop-up-ikkuna. Jos ikkuna oli auennut, Robotti sulki sen, jonka jälkeen prosessi jatkui normaalisti eteenpäin seuraavaan testitapaukseen. Testitapauksen käyttäminen uudessa testissä onnistui vaivatta, koska testitapauksen toiminta oli jo viime viikolla saatu testattua ja testitapaus oli vain siirrettävä oikeaan paikkaan uudessa testissä.

Sain päivällä aloittamani tehtävät toteutettua hyvin ja saavutin samalla päivän aikana muotoutuneen tavoitteen uuden testin ohjelmoimisesta. Osaamiseni kehittyi päivän aikana kertaamalla ja toistamalla jo oppimiani asioita. Täten Robot Frameworkin ohjelmoimiseen löytyi vaihtoehtoisia, uusia ja jopa tehokkaampia toteutustapoja.

Tiistai 20.6.2017

Koska sain viimeisteltyä Telia verkkokaupan käyttöliittymää läpikäyvää testiä, tämän päivän tavoitteena oli aloittaa uuden, mahdollisesti eri web-sivustoa testaavan testiautomaatioharjoituksen ohjelmoiminen. Päivän toisena tavoitteena oli oppia ohjatun perehdytyksen ja itseopiskelun kautta lisää Image Horizon-nimisestä Robot Frameworkin kirjastosta. Image Horizon on graafisen käyttöliittymän automaattiseen testaamiseen tarkoitettu Robot Frameworkin kirjasto, jonka avulla on mahdollista suorittaa kuvantunnistusta (Efficode, 2015). Kirjaston avulla pystytään esimerkiksi vertaamaan, että testattavalta web-sivulta löytyy esimerkiksi Windowsin Snipping Tool-työkalulla otettua kuva vastaava kuva.

Image Horizon-kirjaston avainsanojen käyttöä pääsin harjoittelemaan käytännön kautta ensin ohjatussa perehdytyksessä ja sen jälkeen harjoittelemalla itsenäisesti kuvatunnistusta oman testiautomaation kautta. Kyseisessä testissä Robotti avasi Telia verkkokapan sivut ja etsi Windowsin Snipping Tool-työkalulla otetun kuvan avulla sivustosta halutun elementin. Tässä tapauksessa Robotti etsi sivun vasemmasta ylälaidasta löytyvää Telian logoa, ja klikkasi sitä löydettyään sen.

Ennen siirtymistä päivän tavoitteiden pariin, päätin lisätä eiliseen testiin ylimääräisen testitapauksen, jossa Robotti valitsee alasvetovalikosta vaihtoehtoisen värin tilattavalle tuotteelle. Tämä vei hieman aikaa, koska alasvetovalikon eri elementtien XPath-polkujen löytäminen ja niiden kanssa vuorovaikuttaminen osoittautui odotettua työläämmäksi, vaikka viime viikolla vastaan oli tullut jo useita XPath-polkuihin liittyviä haasteita. Uudet haasteet johtuivat siitä, että alasvetovalikossa olevien elementtien id-tunnukset eivät olleet staattisia vaan dynaamisia. Tunnuksesta löytyi siis osia, jotka muuttuivat joka kerta kun sivu ladattiin uudelleen**.**

Käyttämällä Stack Overflow- ja Robot Frameworkin käyttäjien Google Group-sivustoja apuna, löysimme kesätyökollegani kanssa muutaman eri vaihtoehdon, joiden avulla oli mahdollista löytää tarvittavat elementit niiden id-tunnuksien dynaamisuudesta huolimatta. Yksi näistä vaihtoehtoista oli etsiä haluttu elementti niillä attribuutin osilla, jotka olivat staattisia. Esimerkiksi jos elementin id olisi ”valikko_0000_musta”, josta numerosarja 0000 olisi attribuutin dynaaminen osa, elementti etsittäisiin sen id:n staattisten osien eli ”valikko_” ja ”_musta” avulla. Toinen, yhtä toimiva vaihtoehto oli etsiä elementtiä sen sisältävän tekstin avulla, eli tässä tapauksessa laitteen värivaihtoehdoilla, jotka olivat esillä alasvetovalikossa.

Tämän päivän tavoitteet saavutin hyvin päästessäni oppimaan ja harjoittelemaan täysin uutta asiaa liittyen kuvantunnistukseen Image Horizon-kirjastoa hyödyntämällä sekä myös

kertaamaan jo aikaisemmin oppimiani asioita ja löytämään niistä uusia näkökulmia. Samalla sain myös aloitettua uuden testiautomaatioharjoituksen työstämistä.

Keskiviikko 21.6.2017

Keskiviikon tavoitteena oli perehtyä tarkemmin eilen tutustumaani Image Horizon-kirjastoon GitHubista löytyvän materiaalin (Efficode, 2015) ja itsenäisen harjoittelun avulla. Koska sain ensikosketuksen Image Horizon-kirjastoon vasta eilen, kyseiseen Robot Framework kirjastoon tarkempi perehtyminen oli mielestäni tärkeää siihen liittyvän osaamisen koostuessa vasta muutaman avainsanan hallitsemisesta.

Perehtyessäni tarkemmin Image Horizon-kirjaston materiaaleihin, jatkoin samalla eilen aloittamani kuvantunnistusta hyödyntävän testin työstämistä harjoitellakseni kyseisen kirjaston avainsanojen käyttöä myös käytännössä. Pyrin kokeilemaan uusia avainsanoja, kuten esimerkiksi ”Click To The Left Of”-avainsanaa, minkä avulla Robotti klikkaa ennalta määritellyn kuvan vasemmalle puolelle halutun pikselien määrän verran (Efficode, 2015).

Tämän päiväisessä Robot Framework koulutuksessa kävimme ohjatusti läpi, miten Robot Frameworkilla pystyy laskemaan erilaisten listojen sisältämien alkioiden määrän sekä sen, miten teoriassa Robotti Frameworkilla ohjelmoidaan for-silmukka. For-silmukan avulla pystytään toistamaan halutut koodirivit niin monta kertaa kuin ennalta on määritelty (MatsWichmann, 2017) ja näin ollen sen avulla on mahdollista käydä läpi erikokoisia listoja tai joukkoja ja etsiä niistä esimerkiksi tiettyä arvoa, kunnes arvo löytyy.

Koulutuksen jälkeen pääsin käytännössä harjoittelemaan listan alkioiden laskemista ja for-silmukan tekemistä koulutuksessa käydyn esimerkkikoodin avulla. Varsinaista for-silmukan ohjelmoimista en vielä tänään ehtinyt aloittamaan keskittyessäni ensin listan alkioiden laskemiseen. Listan alkioiden laskeminen onnistui hyvin ”Get Matching Xpath Count”-avainsanalla johon määritellään XPath-polku, joka on kaikilla listan alkiolla sama ja Robotti laskee kaikkien kyseisen XPath-polun omaavien elementtien yhteenlasketun lukumäärän (Robot Framework Foundation, 2017d).

Päivälle asettamani tavoitteet onnistuivat hyvin oppiessani uutta Image Horizon-kirjastosta. Päivän toinen tavoite koski for-silmukkaan ja listan alkioiden lukumäärän laskemiseen liittyvän teorian ja käytännön läpikäymistä sekä itsenäisesti että ohjatusti. For-silmukan käytön harjoittelu käytännössä minun on tarkoitus aloittaa huomenna.

Torstai 22.6.2017

Tavoitteenani oli tänään harjoitella for-silmukka toimintaa käytännössä lisäämällä silmukka viime viikolla ohjelmoimaani Telian verkkokauppaa testaavaan testiin. Tarkoituksenani oli lisätä for-silmukka siihen vaiheeseen tilausprosessia, jossa Robot Framework valitsee tilattavan laitteen lisälaitteelle halutun maksutavan. Näin silmukka kävisi läpi kaikki makutavat Robotin etsiessä haluttua maksutapaa.

Aamupäivällä järjestettiin kokous, jossa sain lisää selvyyttä tuleviin työtehtäviini. Kokoukseen ottivat osaa sekä yrityksen sisäiset että ulkopuoliset testaajat ja kokouksen tavoitteena oli käydä läpi mitä minä ja kesätyökollegani tulemme heinäkuun aikana testaamaan muiden testaajien ollessa lomalla.

Kokouksessa päätetiin, että tulen itse tekemään heinäkuussa erilaisia automatisoituja testejä Robot Frameworkilla, joiden avulla voidaan myöhemmin suorittaa regressiotestausta muutamille eri järjestelmälle web-käyttöliittymän kautta. Regressiotestauksella tarkoitetaan taantumatestausta, jonka tarkoituksena on varmistaa, että testattava järjestelmä toimii edelleen senkin jälkeen, kun järjestelmään on tehty muokkauksia tai korjauksia tai jos kyseinen järjestelmä on ollut vuorovaikutuksessa jonkun muun järjestelmän kanssa (SmartBear Software, 2017). Näissä työtehtävissä pääsen hyvin hyödyntämään kahden ja puolen viikon aikana oppimiani asioita web-sivustojen testaamisesta Robot Frameworkin avulla.

Päivälle asetetun tavoitteen saavuttamiseen liittyvät tehtävät toteutuivat onnistuneesti. For-silmukan ohjelmointi valmiiseen testiin onnistui tavoitteiden mukaisesti. Ennen varsinaista silmukkaa Robotti ilmoittaa ensin listasta löytyvät alkiot merkkijonona ja laskee eilen tekemäni koodin avulla valittavien maksuvaihtoehtojen lukumäärän eilen tutuksi tulleen ”Get Matching Xpath Count”-avainsanan avulla. Valittaessa tilattavan laitteen lisälaitteelle maksutapaa, Robotti käy for-silmukan avulla läpi kaikki eri alasvetovalikosta löytyvät maksuvaihtoehdot ja ilmoitti kun haluttu maksuvaihtoehto löytyy valikosta.

Kun haluttu maksuvaihtoehto löytyi listasta, Robotti valitsi kyseisen alkion, joka sisältää ennalta määritellyn merkkijonon. Lopuksi ennen silmukasta poistumista ja seuraavaan testitapaukseen siirtymistä, Robotti kirjoitti konsoliin myös valitun maksuvaihtoehdon ja tämän järjestysnumeron listan alkioista. Listan alkioiden järjestysnumeroita tutkittaessa huomasin, että tässä tapauksessa alkioiden numerointi alkoi nollasta eikä numerosta yksi. Eli tällöin esimerkiksi listan neljäs alkio kantoikin järjestysnumeroa kolme, eikä numeroa neljä.

Päivän oppiminen for-silmukan ohjelmoimiseen liittyen tapahtui hyödyntämällä eilisen Robot Framework-koulutuksen esimerkkimateriaaleja, tiimin yhteistä testiautomaatiota koskevaa versionhallintaa sekä itse kokeilemalla ja omista virheellisistä koodeista oppimalla. Tiimin yhteinen versionhallinta toimi erittäin hyvänä oppimateriaalina tähänkin harjoitteluun, koska versionhallinnasta löytyi kattavat materiaalit Robot Frameworkin eri osa-alueisiin liittyen.

Perjantai 23.6.2017

Tästä päivästä ei ole erillisiä raportointimerkintöjä, koska kyseinen päivä oli Juhannusaatto ja näin ollen vapaapäivä töistä.

3.2.1 Viikkoanalyysi 2

Toisen raportointiviikon aikana Robot Frameworkiin liittyvä osaamiseni kehittyi taas eteenpäin uusiin aihealueisiin, kuten kuvantunnistukseen ja for-silmukkaan, tutustumisen kautta. Koska tämän viikon työskentely tapahtui hyödyntämällä mahdollisimman paljon viime viikolla aloittamaani verkkokaupan testiautomaatiota, sain samalla myös kerrattua aiheita, joihin olin päässyt aikaisemmin tutustumaan. Aktiivisen kertaamisen kautta sain opittua lisää jo tutuista aiheista ja näin sain myös vakiinnutettua omaa työskentelytapaani Robot Frameworkin parissa**.**

Uusiin aiheisiin perehtymisessä auttoivat valmiit esimerkit ja materiaalit, jotka olivat saatavilla sekä internetistä että tiimimme testiautomaatiosta vastaavien henkilöiden yhteisestä versionhallinnasta. Hankalan tilanteen tullen sain tarvittaessa apua myös kysymällä oman tiimini jäseniltä ja pyrin pohtimaan ongelmatilanteita myös yhdessä kesätyökollegani kanssa. Asioiden yhdessä käynti helpotti oppimista mielestäni melko paljon, koska yhdessä pystyimme keskustelemaan ja pohtimaan eteen tulleita kysymyksiä ja pystyimme hyödyntämään toistemme osaamista, näkökulmia ja ongelmanratkaisukykyjä.

Tällä viikolla saimme ohjatusti käytyä läpi loputkin niistä Robotin osa-alueista, joiden perusteet osaamalla minun on mahdollista aloittaa työtehtäviini kuuluvien testien ohjelmoiminen. Aktiivisen itsenäisen toiminnan ja omien testiautomaatioharjoitusten avulla olen päässyt tehokkaasti harjoittelemaan käytännön kautta miten Robot Frameworkin eri ominaisuudet toimivat ja näiden eri harjoitusten kautta olen myös saanut tehtyä itselleni erilaisia valmiita pohjia, joita pystyn suoraan hyödyntämään tulevissa web-testiautomaatioissa. Nämä valmiit pohjat toimivat myös hyvänä muistilistana, joista näen suoraan miten mikäkin Robotin ominaisuus toimii ja mikä on paras tapa hyödyntää niitä. Tässä minua on aut-

tanut se, että olen kommentoinut tekemääni Robotin koodia selkeästi ja mahdollisimman aktiivisesti.

Tekemistäni kommenteista olen pystynyt suoraan näkemään muistin virkistykseksi mitä mikäkin koodin osa tekee. Jätin samalla kommentteihin näkyviin näin aluksi myös ratkaisuja, jotka eivät toimineet. Näin näen suoraan, mikä tapa on toiminut missäkin tilanteessa ja mikä ei välttääkseni turhan yrityksen ja erheen aiheuttaman toiston uusia testejä ohjelmoidessani.

Tämän toimintamallin opin alun perin ammattikorkeakoulussa ohjelmoinnin kursseilla, joissa kehotettiin käyttämään kommentointia ja lokimerkintöjä mahdollisten virheiden kartoittamiseen. Selkeällä kommentoinnilla pystyi hyvin myös kuvaamaan niin itselle kuin ulkopuoliselle lukijallekin mitä missäkin koodin vaiheessa tapahtuu. Kun testiautomaatiosta vastaava kollegani aloitti Robot Frameworkin kouluttamisen minulle, hän kehotti myöskin pitäytymään tässä toimintamallissa.

Tämä toimintamalli osoittautui hyvin hyödylliseksi varsinkin silloin, kun tekemäni koodi ei toiminut. Ylimääräisten lokimerkintöjen avulla pystyin tehokkaammin etsimään missä vaiheessa koodia virhe oli tapahtunut. Samasta toimintatapaan liittyen Richard Bradshaw toteaa blogissaan (2016), että automatisoiduille testeille tulisi antaa ”oma ääni” lokimerkintöjen ja kommenttien avulla, mikä nopeuttaa huomattavasti virheiden etsimistä ja helpottaa ajettavien testien seuraamista.

Työtehtävissäni määriteltyjen testiautomaatioiden tekemiseen vaadittava perusosaaminen on mielestäni noin kahden ja puolen viikon harjoittelun jälkeen melkein hallinnassa. Ennen varsinaisen automatisoinnin aloittamista minun on kuitenkin tehtävä itselleni tutuiksi vielä testattavat prosessit eli ne vaiheet, joiden kautta kyseiset prosessit etenevät. Kanerin, Bachin ja Pettichordin (2002, s.17) mukaan testataksesi jotain hyvin, oli kyseessä kokonainen järjestelmä tai yksittäinen ominaisuus, sinun täytyy työskennellä sen kanssa ja sitä täytyy tutkia jatkuvasti oppien ja kokeillen. Testattavien prosessien ja järjestelmien tunteminen onkin edellytys testiautomaatioiden ohjelmoimiselle. Minun on siis tiedettävä mitä tulen testaamaan ja automatisoimaan. Jos en itse tiedä miten esimerkiksi uuden laitteen tilausprosessi etenee, ei voi sitä myöskään automatisoida.

Tällä hetkellä omat työtehtäväni tulevat tänä kesänä liikkumaan eri tuotteiden ja palveluiden, kuten laajakaistayhteyden ja uuden mobiililaitteen, tilausprosessien automatisoinnin ympärillä. Uskon, että erilaisten tuotteiden tilausprosessit ovat kuitenkin yleisellä tasolla melko samanlaisia. Omat haasteensa testaamiseen tuovat kuitenkin ne, miten testattava

prosessi etenee eri järjestelmissä ja kenen näkökulmasta tilausprosessia tarkastellaan. Näkökulma, asiakkaan tai Telian työntekijän, vaikuttaa esimerkiksi järjestelmän käyttöön liittyviin oikeuksiin, jolloin järjestelmien käyttöliittymät voivat vaihdella. Nämä seikat on otettava huomioon ennen testien ohjelmoinnin aloittamista, minkä takia on entistä tärkeämpää tietää, mitä testataan ja missä järjestelmässä.

Tällä viikolla ongelmia aiheuttivat pääasiassa uusien aiheiden, kuten kuvantunnistuksen ja for-silmukan, teorian soveltaminen käytännössä. Näitä vastaan tulleita ongelmia lähdin lähestymään etsimällä ensin mahdollisia ratkaisuja valmiista materiaaleista ja esimerkeistä. Tämän jälkeen lähdin kokeilemaan löytämääni mahdollista ratkaisua käytännössä.

Kuten ennalta arvelinkin, lähes mikään vastaan tulleista ongelmista ei ratkennut ensimmäisellä yrityksellä vaan vasta useamman yrityksen kautta. Tämä oli kuitenkin hyvä asia, koska yrityksen ja erehdyksen kautta sain kartoitettua läpi lisää eri ratkaisu- ja toimintatapoja eri tilanteisiin liittyen. Esimerkiksi tietyn web-elementin XPath-polkua etsiessäni for-silmukkaa varten, hyödynsin jo aikaisemmin käyttämääni toimintatapa. Samalla kirjasin myös lähes kaikki kokeilemani eri vaihtoehdot ylös ja merkkasin mitkä niistä toimivat ja mitkä eivät.

Erityisesti ongelmia minulle tuotti kuitenkin for-silmukka. Minulla on ollut hankaluuksia sen kanssa myös eri ohjelmointikielten, kuten Javan, kohdalla. Ammattikorkeakoulussa opin erilaisten harjoituksien ja kotitehtävien kautta ymmärtämään for-silmukan toimintaa Java-ohjelmointikielessä tarkemmin. Opin myös käyttämään for-silmukkaa, mutta tehtävien ja kurssien jälkeen kyseisen funktion toiminnasta jäi silti hieman epävarma olo.

Harjoitellessani tämän viikon loppupuolella for-silmukan luomista Robot Frameworkilla, huomasin, että olin vihdoin saanut tarvittavan selkeyden for-silmukan toiminnasta. Vaikka for-silmukan ohjelmointi Robotilla eroaa sen ohjelmoinnista Javalla ja vaikka kyseessä on melko pieni saavutus, olin silti tyytyväinen. Sain opittua uusien harjoitteiden avulla tästä aiheesta lisää sen ollessa ollut aikaisemmin minulle hieman epäselväksi jäänyt aihe.

3.3 Seurantaviikko 3

Maanantai 26.6.2017

Päivän tavoitteena oli kerrata vielä lisää Robot Frameworkin for-silmukkaan liittyvää käytäntöä ja teoriaa sekä mahdollisesti myös ohjelmoida viime viikon aikana ohjelmoitua silmukkaa eteenpäin. Päivän toisena tavoitteena oli saada lisää selvyyttä omiin tuleviin työ-

tehtäviini aamulla järjestetyn kokouksen kautta. Aamuinen kokous oli jatkoa viime viikkoiselle kokoukselle, jossa käsiteltiin mahdollisia automatisoitavia testejä, joita voisin ohjelmoida heinäkuussa.

Saavutin päivän tavoitteeni kertaamalla Robotin for-silmukan teoriaa ja harjoittelemalla lisää sen käyttöä. Yritin myös lähteä kehittämään silmukkaa eteenpäin niin, että valittaessa Telian verkkokaupasta uutta matkapuhelinta, Robotti kävisi for-silmukan avulla läpi laitteen värivaihtoehdot ja valitsisi niistä sen vaihtoehdon, jota on varastossa saatavilla. Tämä edellyttäisi sitä, että Robotti kävisi ensin kaikki listan alkiot läpi ja tarkistaisi jokaisen alkion kohdalla, että sivulta löytyisi teksti, joka ilmoittaisi, että kyseinen värimalli olisi saatavilla. Jos haluttua värivaihtoehtoa ei olisi saatavilla, Robotti valitsisi listasta seuraavan varastosta löytyvän värivaihtoehdon. Tätä ominaisuutta en kuitenkaan saanut valmiiksi, sen osoittauduttua työläämmäksi ja haastavammaksi kuin oletin. Oppimisen kannalta tämä harjoittelu oli kuitenkin hyväksi, koska sain taas lisää käsitystä siitä, miten for-silmukka toimii ja miten sitä tulisi käyttää. Pyrin palaamaan tämän harjoituksen pariin, kun vain tulevilta työtehtäviltäni kerkeän, jotta se ei jäisi kesken.

Toinen tälle päivälle asettamani tavoite tuli myös saavutettua. Aamupäivällä järjestetyssä kokouksessa käytiin läpi, mille järjestelmille tulisin automatisoimaan erilaisia testejä. Heinäkuussa minun tulee automatisoida Telian verkkokaupan, erään myyntijärjestelmän sekä asiakkaan omien sivujen testisivustojen erilaisia tilaus- ja muutosprosesseja. Näitä automatisoituja testejä pystyttäisiin mahdollisesti käyttämään malleina tulevissa regressiotestauksissa. Kokouksessa oli puhetta myös eri tilauksien poistoprosessien automatisoimisesta eri päätejärjestelmistä, mutta tarkoituksena on, että nyt kesätyösuhteeni aikana keskittyisin vain tilaus- ja muutosprosessien automatisoimiseen.

Järjestimme iltapäivälle myös erillisen kokouksen, jossa kävimme pintapuoleisesti läpi kustakin testattavasta järjestelmästä yhdet esimerkit eri tilausprosesseista. Tutustuimme myös myyntijärjestelmän ja asiakkaan omien sivujen käyttöliittymään, jotka eivät olleet minulle entuudestaan tuttuja. Uskon, että ainakin verkkokaupan testisivun tilausprosessin automatisoimisessa helpottaa se, että olen omatoimisesti jo tutustunut kyseiseen sivustoon omien testiautomaatioharjoituksieni kautta viikoilla yksi ja kaksi.

Alkavia automatisointitehtäviä varten joudun perehtymään entistä tarkemmin verkkokaupan, myyntijärjestelmän sekä asiakkaan omien sivujen testi-instansseihin ja niiden kautta tehtäviin tilauksiin. Näiden järjestelmien ja tilausprosessien läpikäymisestä sekä niistä oppimisesta muodostuvatkin hyvät tavoitteet tuleville päiville ja pyrin huomenna perehtymään ainakin verkkokaupan ja myyntijärjestelmän kautta tehtäviin tilauksiin.

Tiistai 27.6.2017

Asetin eilen tämän päivän tavoitteeksi perehtyä eri tilausprosesseihin ainakin verkkokaupassa ja myyntijärjestelmässä. Näihin tilausprosesseihin kuuluivat kiinteän laajakaistan (kaapeli ja valokuitu), matkapuhelimen, puhelinliittymän sekä sähköisen lisäpalvelun VPN:n tilaaminen. Tarkoituksenani oli samalla kirjoittaa eri tilausprosesseista selkeät etenemisvaiheet ylös työn automatisoinnin helpottamiseksi.

Käymällä vaihe kerrallaan eri tilausprosesseja läpi eri järjestelmissä, sain saavutettua päivälle asettamani tavoitteet ja opittua paljon uutta näistä tilausprosesseista sekä käytettävistä järjestelmistä. Päivän tavoitteista jäi ainoastaan puuttumaan laajakaistan tilaamisen läpikäynti verkkokaupassa. Jostain syystä laajakaistapakettia tilattaessa, testisivusto antoi virheilmoituksen ”Tietojen haku epäonnistui, yritä uudelleen”, emmekä päässeet kesätyökollegani kanssa tästä tämän päivän aikana eteenpäin.

Käydessämme tilausprosesseja läpi opin samalla myös lisää siitä, miten varsinkin myyntijärjestelmä ja sen käyttöliittymä toimivat sekä miten yrityksen työntekijä, esimerkiksi asiakaspalvelija, pystyy tekemään asiakkaalle uuden tilauksen. Opin myös enemmän verkkokaupan käytöstä eri tilausprosessien kautta. Aikaisempi tuntemukseni verkkokaupan käyttöön perustui tuotannossa olevaan versioon ja pelkästään uuden mobiililaitteen tilausprosessiin.

Käydessämme kesätyökollegani kanssa eri tilausprosesseja vaihe vaiheelta läpi kirjasimme ne samalla ylös Excel-tiedostoon. Tällaisesta dokumentoinnista on paljon hyötyä, kun aloitamme testiautomaatioiden kirjoittamisen, koska se toimii samalla myös testin kehitysjonona, josta lukija näkee suoraan mitä vaiheita ja toimintoja tietyn tilauksen eteenpäin vieminen vaatii ja mikä on minkäkin toiminnon lopputulos. Kehitysjonoa käytetäänkin varsinkin ketterässä tuotekehityksessä, jossa sen tarkoituksena on listata kaikki tuotteelle vaadittavat ja suunnitellut ominaisuudet (Scrum.org, 2017). Tämä malli soveltui hyvin myös tekemääni automatisointiin ja kehitysjonosta pystyn esimerkiksi helposti seuraamaan myös missä vaiheessa kunkin testiautomaation tekeminen on.

Eräs kollegani toi aamupäivästä tietoisuuteni Xmind-nimisen, ilmaisen työkalunsovelluksen. Xmind on ajatuskarttatyökalu, jonka avulla käyttäjä pystyy luomaan helposti erilaisia ajatuskarttoja (XMind, 2017). Perehtyessäni ja opetellessani kyseisen sovelluksen käyttöä, yllätyin positiivisesti siitä, kuinka helposti sillä pystyi luomaan visuaalisesti miellyttäviä ja selkeitä ajatuskarttoja. Yhdeksi huomisen päivän tavoitteena asetin muutamien läpikäy-

tyjen tilausprosessien ja niiden eri vaiheiden mallintamisen Xmindiin katsoakseni, kuinka hyvin se soveltuu eri tilausprosessien kuvaamiseen Excelin ohella.

Keskiviikko 28.6.2017

Päivän päätavoitteena oli jatkaa tilausprosessien tutkimista ja mallintamista. Saavutin tavoitteeni hyvin, koska saimme tänään kollegani kanssa käytyä läpi muun muassa TV-palvelun ja sen eri kanavapakettien tilausprosessit. Saimme aloitettua myös laitevakuutuksen tilaamisen läpikäymistä. Ongelmaksi nousi kuitenkin tilanne jossa, vakuutus tilattiin itsenäisenä palvelunaan ja myyntijärjestelmä pyysi IMEI-testitunnuksia, jota meillä ei ollut hallussamme. Tätä ja muutamaa muuta tilausprosessia lukuun ottamatta, testattavat tilausprosessit oli nyt käyty läpi. Päivän aikana opin taas lisää eri tilausprosesseista ja järjestelmien käytöstä. Kävimme päivän aikana myös tarkemmin läpi versionhallinta työkalun GitHubin konsolikäyttöliittymän GitBashin käyttöä erilaisten konsolikomentojen kautta.

Xmindin-työkalua hyödyntämällä sain päivän aikana mallinnettua selkeästi kaikki eilen läpikäymäni tilausprosessit kopioimalla tilauksien eri vaiheet ja niiden oletetut lopputulokset suoraan kehitysjonosta. Näin ollen saavutin eilen Xmindin käyttöön asettamani tavoitteet tilausprosessien mallintamisesta. Lopputuloksena syntyivät hyvin selkeät ajatuskartat ja tilausprosessien mallintaminen oli kaiken kaikkiaan hyödyllinen oppimisprosessi niin itse työkalun käytön kuin myös projektihallinnan suunnittelun kannalta.

Torstai 29.6.2017

Tämän päivän tavoitteeni liittyivät GitHub-versionhallintatyökalun ja sen moduulin GitBashin käyttöön. Tavoitteenani oli kerrata GitHubin ja sen konsolikäyttöliittymän komentoja sekä kopioida käyttämämme versionhallinnan päähaaralta kaikki tarpeellinen sisältö omalle työasemalleni. Päivän tavoitteet liittyivät GitHubin käyttöön. Aikaisemmat kokemukseni GitHubista olivat koulusta viime vuoden syksyltä, joten näin kertauksen olevan tarpeen.

Testiautomaatioita työstäessämme emme kuitenkaan suoraan käytä GitHubia. Käytössämme on Deveo-niminen versionhallinta-alusta, joka on eräs selainpohjainen, graafinen käyttöliittymä GitHubiin. GitHubin tavoin, Deveo sisältää työstettävät projektit ja niiden hakemistot. Deveon ja GitBashin ollessa eräät työni tärkeimmistä työkaluista, näin hyvin tärkeäksi tehdä niiden toiminta mahdollisimman selväksi. Käyttäessäni aikaisemmin GitHubia koulussa, käytin pääsääntöisesti graafista käyttöliittymää. Tästä johtuen, konsoli-

komentojen opettelussa ja kertaamisessa meni hetki, ennen kuin sain ne taas muistumaan mieleeni.

Hakiessani versionhallinasta kaiken tarvittavan materiaalin uusien testiautomaatioiden työstämisen helpottamiseksi, teimme samalla kesätyökollegani kanssa yksinkertaisen kaavion liittyen GitBashin käyttöön. Kaaviosta lukija näkee suoraan eri vaiheet ja komennot, joilla hän esimerkiksi saa vietyä tekemänsä muutokset paikalliselta työasemalta ulkoiseen versionhallintaan eli tässä tapauksessa Deveoon. Oppisen kannalta on erittäin hyvä, että käytössämme on GitBash-konsolinäkymä eikä pelkästään graafinen käyttöliittymä.

Kun sain opeteltua GitBashin käyttöä ja mallinnettua loput eilen läpikäymäni tilausprosessit Xmind-ohjelmaan, aloitin iltapäivällä työkuvaani kuuluvien testiautomaatioiden tekemisen. Lähdin liikkeelle uuden mobiililaitteen tilauksesta, joka tehdään Telian verkkokaupasta. Tämä tilausprosessi oli minulle jo entuudestaan tuttu, koska kyseessä oli sama tilausprosessi johon olin omatoimisen harjoittelun kautta perehtynyt jo raportointiviikoilla yksi ja kaksi. Omien valmiiden harjoittelutestien hyödyntämisen lisäksi sain käyttööni kaikki minua testiautomaatiossa opettaneen kollegan tekemät materiaalit ja esimerkit. Samalla sain käyttööni vastikään luodun, valmiin hakemiston pohjaksi omien testiautomaatioiden luontia varten. Uuden testin automatisoinnista kehittyi päivän uusi tavoite ja sainkin päivän aikana aloittamani testiautomaation tilausprosessin viimeistä vaihetta eli tilauksen maksamista verkkopankissa lukuun ottamatta valmiiksi.

Perjantai 30.6.2017

Tämän perjantain tavoitteena oli käydä vaihe kerrallaan läpi uuden Jenkins-työn luonti osatakseni itse luoda niitä, jos myöhemmin kesällä tulee tarve ajaa tekemiäni testiautomaatioita myös Jenkinsin kautta. Päivän toisena tavoitteena oli saada eilen aloittamani uuden mobiililaitteen tilauksen automatisointi valmiiksi. Kyseisestä testistä puuttui enää maksun suorittaminen verkkopankin testisivulla.

Olimme aikaisemmin käyneet kollegani kanssa pintapuoleisesti läpi, miten uusi Jenkins-työ luodaan, mutta tänään kävimme sen tarkasti vaihe vaiheelta läpi. Oppimisen tehostamiseksi kirjoitin itselleni kyseiset vaiheet ylös. Perehdytyksen aikana opin ne vaiheet, joita tarvitaan uuden Jenkins-työn luomiseksi ja omien testiautomaatioiden ajamiseksi Jenkinsin kautta Linux- tai Windows-virtuaalikoneilla. Pyysin itse omatoimisesti tämän palaverin järjestämistä, koska kun aikaisemmin kävimme Jenkinsin käyttöä läpi, kyseessä oli melko nopea ja pintapuoleinen katsaus kyseiseen työkaluun. Koska työnkuvaani liittyen Jenkins on yksi työkaluistani Robot Frameworkin ja GitBashin ohella, oppisen kannalta oli parempi

järjestää lisää aikaa työkalun perusteellisempaan tutustumiseen. Olin hyvin tyytyväinen tähän perehdytykseen, koska sen myötä Jenkinsin osaamiseni kehittyi tavoittelemaani pisteeseen. Tarvittavien perustoimintojen ollessa selkeitä, myös uuden oppiminen ja teorian läpikäyminen on huomattavasti helpompaa pohjatietojen ollessa kunnossa.

Myös päivän toinen tavoite tuli saavuttua, kun sain viimeistelyä uuden mobiililaitteen tilaamisen automatisoitua. Tilausprosessi kattaa verkkokaupan sivulle siirtymisen, määritellyn laitteen valitsemisen ja siirtämisen ostoskoriin, asiakastietojen syöttämisen ja laitteen maksamisen testiverkkopankissa. Vaikka tämä tilausprosessi on nyt valmis, tulen todennäköisesti palaamaan siihen vielä koodin hiomisen ja avainsanojen siistimisen yhteydessä Robot Frameworkin ohjelmointitaitojeni kehittyessä.

Päivän päätteeksi aloitin seuraavan tilausprosessin automatisointia, jossa mobiililaite tilataan verkkokaupasta, mutta laitteen maksutavaksi valitaan tällä kertaa osamaksu eli rahoitus. Tämän uuden testiautomaation aloittaminen kävi helposti hyödyntämällä aiempaa automaatiota pohjana. Kertamaksulla tilattavan mobiililaitteen tilausprosessi oli täysin sama kuin kertamaksulla tilattavan tuotteen lukuun ottamatta maksutapaa, verkkopankin kautta tunnistautumista ja rahoitukseen liittyvän luottarajan valitsemista.

En kuitenkaan saanut työstämääni testiautomaatiota valmiiksi tämän päivän aikana, koska tässä testissä ongelmaksi nousi luottorajan valitseminen. Yrittäessäni valita testikäyttäjälle luottorajaa, kävikin ilmi, että kyseisen testitunnuksen testiluottotiedot olivat muiden aikaisemmin suoritettavien testien takia epäkunnossa. Tästä syystä uuden testiluoton myöntäminen ei käyttämälleni testikäyttäjälle olut mahdollista. Jouduin jättämään testin automatisoinnin tähän vaiheeseen odottamaan, koska luottotiedoiltaan toimivia testitunnuksia ei ollut vielä saatavilla. Muuten tässä tilausprosessissa ei ollut enää montaa vaihetta automatisoimatta.

3.3.1 Viikkoanalyysi 3

Kolmannen raportointiviikon aikana ehti tapahtua paljon varsinaisten työtehtävien alkaessa. Odotin tätä viikkoa hyvin innoissani, koska pääsisin vihdoin hyödyntämään ensimmäisten viikkojen aikana keräämiäni Robot Framework-taitoja varsinaisissa työtehtävissäni. Tämän viikon aikana oppiminen Robottiin liittyen tapahtui pääasiassa kokeilun ja kertauksen kautta.

Uusia testiautomaatioita ohjelmoidessani kävin läpi niin vanhoja, kuin uusiakin materiaaleja läpi toiminta- ja työskentelytapojeni vakiinnuttamiseksi. Tähän liittyy esimerkiksi Robotin

koodin rakenteen ja laadun parantaminen käyttämällä yhteisestä hakemistosta löytyviä ja kollegoiden kanssa ennalta sovittuja ohjelmointiperiaatteita. Tällaisia olivat esimerkiksi isojen alkukirjainten käyttö sekä avainsanojen että testitapauksien nimeämisessä ja niiden selkeä nimeäminen niin, että nimestä saa heti käsityksen siitä, mitä kukin avainsana ja testitapaus tekevät. Näitä käytännön toimintatapoja pääsin hyödyntämään suoraan aloittaessani työnkuvaani kuuluvien testiautomaatioiden tekemisen. Samalla sain vakiinnutettua tätä toimintatapaa varsinkin jatkossa ohjelmoitavia testejä varten.

Olin jo aikaisemmissa oma-aloitteisissa Robot Framework-harjoituksissani ottanut vahvasti toiminnassani mallia valmiissa materiaaleissa ja esimerkeissä käytetyistä toimintamalleista. Silloin Robotin osaamiseni ei kuitenkaan ollut samalla tasolla kuin nyt, joten jouduin tekemään koodin rakenteen ja laadun kanssa kompromisseja, jotta sain testin toimimaan haluamalla tavalla. Esimerkiksi ensimmäisissä ohjelmoimissa testeissäni käytin vain tietyn kirjaston avainsanoja ja yritin saada kaikki testini toimivaan vain kyseisiä avainsanoja hyödyntämällä, koska en vielä täysin ymmärtänyt Robotin logiikkaa.

Jos olisin työskentelyssäni jatkanut tämän toimintatavan hyödyntämistä, vastaani olisi ennemmin tai myöhemmin tullut tilanne, johon minulle tutut avainsanat eivät olisikaan riittäneet. Kuten Kaner, Bach ja Pettichord (2002, s.115) kirjoittivat, en olisi tällöin pystynyt viemään testiautomaatioitani eteenpäin, koska tilanteeseen sopivia avainsanoja ei käyttämässäni kirjastossa olisi ollut olemassa. Saadessani jo muutamien päivien sisällä pikkuhiljaa parempaa kokonaiskuvaa Robotista ja sen toiminnasta, tämä työskentelytapa kuitenkin hioutui pois ja uskalsin lähteä hyödyntämään muitakin kuin vain yhtä Robot Frameworkin kirjastoa osaamiseni kasvaessa jatkuvasti.

On ollut tyydyttävää huomata, että oma osaamiseni Robot Frameworkin käytössä on kehittynyt ja on kehittymässä eteenpäin haluamaani suuntaan. Esimerkiksi uusien testien ohjelmoiminen on ollut huomattavasti helpompaa oppimisen kehittyessä ja olen pääsyt itse harjoittelemaan etukäteen niitä asioita, joita tulen nyt työtehtävissäni tekemään. Samalla minulle on kertynyt parempi kokonaiskäsitys siitä miten eri avainsanat toimivat, miten Robotin koodi rakentuu ja miten koodi pidetään laadullisesti kunnossa.

Viikon oppimiseni ei painottunut pelkästään Robot Frameworkiin vaan opin täysin uusia asioita Jenkinsiin ja GitBashiin liittyen. Työtehtävieni ja oman oppimiseni kannalta oli mielestäni hyvin tärkeää päästä käymään tarkemmin läpi näiden työkalujenohjelmien käyttöä, koska Robotin ohella tulen käyttämään varsinkin GitBashia hyvin paljon. Vaikka en välttämättä tule kesätyösuhteeni aikana käyttämäänkään Jenkinsiä niin paljoa kuin alun perin oli tarkoitus, on silti ollut hyvä perehtyä sen toimintaan jatkoa ajatellen.

Kartoittaessani viikon aikana eri tilausprosesseja testien ohjelmointia varten opin samalla miten kyseiset prosessit etenevät vaihe vaiheelta ja opin myös käyttämään niitä järjestelmiä, jotka eivät vielä olleet minulle täysin tuttuja. Koska minun tulee mahdollisesti myös tarkistaa, että automatisoidut testitilaukset ovat menneet eteenpäin niitä käsitteleviin järjestelmiin ja mahdollisesti myös poistaa näitä testitilauksia, on hyvä käydä jo tässä vaiheessa läpi näiden päätejärjestelmien käyttöä. Tällainen järjestelmä on esimerkiksi CRM-järjestelmän, josta muun muassa myyntijärjestelmän tilauksia pystyy hallinnoimaan ja poistamaan.

Tällä viikolla perehdyin myös Xmind-työkalun käyttöön ja omasta mielestäni tilausprosessien kuvaaminen Exceliä visuaalisemmalla työkalulla helpottaa huomattavasti vastuullani olevien testiautomaatioprojektien ylläpitoa ja hallintaa. Rikard Edgren kirjoittaa (2012, s.20), että testien esittäminen visuaalisesti voi tuoda mukanaan uusia ajatuksia ja ymmärryksiä ja jotkut asiat onkin tehokkaampaa esittää kuvin. Tämän huomasin myös itse, koska erilaisen värien, muotojen, muistiinpanotapojen, ikonien ja mallien avulla käyttäjä näkee selkeästi mitkä vaiheet ovat jo automatisoitu, mitkä ovat vielä tekemättä, mitkä eivät toimi odotetusti ja missä on muuten vielä jotakin huomioitavaa. Puumaisen rakenteen ansiosta käyttäjä pystyy keskittymään kerrallaan yhteen vaiheeseen tai tarkastelemaan projektin päätason ”oksia” kokonaiskuvan hahmottamiseksi.

Varsinkin käyttäjille, joille asioiden visuaalinen kuvaaminen helpottaa niiden hahmottamista, Xmindin kaltainen työkalu on hyvin hyödyllinen ja tutustumisen arvoinen. Erilaisia työn dokumentointiin tarkoitettuja työkaluja hyödyntämällä ja niihin tutustumalla olen oppinut myös sen, että käyttäjän ei tulisi liikaa takertua pelkästään yhteen työkaluun. On kuitenkin olemassa erilaisia samaan käyttötarkoitukseen tarkoitettuja työkaluja, joista jotkut sopivat tiettyihin käyttötarkoituksiin toisia paremmin.

Eniten ongelmia tämän viikon aikana tuottivat puuttuvat testidatat, joiden puuttumisen takia en päässyt kartoittamaan kaikkia tilausprosesseja läpi, koska en saanut vietyä tiettyjä prosesseja loppuun asti vaan. Esimerkiksi perjantaina mobiililaitteen tilausta rahoituksella ei voinut viedä loppuun asti, koska käytettävän testitunnuksen luottotiedot eivät olleet kunnossa ja näin minkäänlaista luottorajaa ei voitu määrittää.

Valitettavasti testidataan liittyvät ongelmat eivät tämän viikon aikana selvinneet. Kyselin tarvittavista testidatoista, mutta tarpeellista testidataa ollut sillä hetkellä saatavissa. Niin kauan kuin tiettyihin prosesseihin tarvittava testidata puuttuu, minun täytyy vain työstää kyseisiä tilausprosesseja niin pitkälle kuin mahdollista, jättää ne siihen vaiheeseen odot-

tamaan ja siirtyä työstämään seuraavia tilausprosesseja. Tämä onkin työn sujuvuuden ja etenemisen kannalta tärkeää, koska ongelma ei ole itsestäni riippuvainen. Voin tietysti aina kysyä testidatan perään, mutta liika hoputtaminen ei välttämättä vie asiaa eteenpäin haluamallani tavalla. Minun on kuitenkin otettava huomioon se, että ihmisillä on eri määrä työtehtäviä hoidettavanaan. Oma ajatusmaailma ja toiminta täytyy siis suhteuttaa sen mukaan ja omia tehtäviä on järjesteltävä niiden tärkeyttä ja aikataulua silmällä pitäen.

Ongelmanratkaisuun liittyen aloittaessani varsinaisten työtehtävien tekemisen, tulen niin sanotusti hyppäämään suoraan syvään päätyyn, kun testiautomaatiosta vastaava kollegani jää koko heinäkuuksi lomalle. Vaikka sainkin käyttööni hänen valmistelemansa esimerkit ja mallipohjat versionhallinnasta omien testien luomiseksi, kollegan poissaolo pakottaa aktiiviseen oma-aloitteisuuteen. Koska paikalla ei ole muita keneltä kysyä Robot Frameworkiin ja sen toimintaan liittyviä kysymyksiä, ongelmat on ratkaistava itse kaikkia saatavilla olevia lähteitä hyödyntämällä. Tämä on mielestäni hyvä asia, koska tällaisen itsenäisen työskentelyn ja ongelmaratkonnan kautta oma osaaminen tulee varmasti kehittymään.

3.4 Seurantaviikko 4

Maanantai 3.7.2017

Aloitin tämän viikon pyrkimällä jatkamaan uusien testien ohjelmointia. Päivän aikana tarkoituksenani oli saada valmiiksi ainakin testi uuden mobiililiittymän tilaamiseksi verkkokaupasta. Tämän lisäksi tarkoituksena oli ajaa yksi omista testiautomaatioistani Jenkinsin kautta harjoituksen vuoksi, jos se olisi mahdollista.

Päivän tavoite ei tullut täysin saavutettua, koska heti aamusta törmäsin ongelmiin GitBashin kanssa. Jostain syystä hakiessani uusimpia korjauksia ja muutoksia työhakemistomme pääoksalta, GitBashin konsoli ilmoitti tiedostojen yhdistämisongelmista. Yritin muutaman tunnin korjata kyseistä ongelmaa, joka koski ymmärtääkseni Xmind-tiedostoa, jossa olen ylläpitänyt tilausprosessien automatisoinnin etenemistä.

Ongelma ei kuitenkaan useammasta eri yrityksestä huolimatta ratkennut. Lopulta päädyin luomaan täysin uuden haaran työhakemistoomme, jonka alustin pääoksan tiedostoilla. Näin sain haettua uudet korjaukset ja muutokset uudelle oksalleni ilman ongelmia. Koska olin viime viikon aikana työskennellyt jo uusien testiautomaatioiden parissa, minulla oli työasemallani Robot Framework-tiedostoja ja koodirivejä, joita pääoksalta ei löytynyt. Nämä aikaisemmin tekemäni työt sain vietyä manuaalisesti uuden haarani kansioraken-

teeseen ja lopulta vein kaiken yhdessä GitBashin kautta Deveoon. Tekemäni ratkaisu kohtaamaani ongelmaan ei ollut välttämättä paras mahdollinen, mutta näin säästin aikaa ja pääsin jatkamaan työskentelyä.

Päästyäni jatkamaan töitä sain aloitettua uuden testin ohjelmoimista, jossa Robotti tilaa verkkokaupasta uuden mobiililiittymän. Työ eteni sujuvasti, koska pystyin käyttämään hyväkseni aikaisemmin tekemiäni verkkokaupan testejä liittyen uuden mobiililaitteen tilaukseen. Ongelmaksi nousi käyttäjän tunnistautuminen Nordean Tupas-tunnistautumisen kautta. Kävi ilmi, että käyttämilläni testitunnuksilla oli tehty maksimimäärä tunnistautumisia, joita tunnuksella voi kuukaudessa tehdä. Tiistain aikana minun täytyykin selvittää, onko mahdollista ottaa käyttöön muita testitunnuksia, jotta saan vietyä aloittamani automaation loppuun.

Tiistai 4.7.2017

Saatuani eilen automatisoitua uuden mobiililiittymän tilausprosessin vaiheeseen, josta en pysty etenemään ilman toimivia testitunnuksia, päätin siirtyä seuraavaan verkkokaupan kautta tehtävään tilaukseen. Tekemällä näin pystyn palaamaan keskeneräisiin töihin myöhemmin ja viimeistellä ne saatuani käyttööni uudet, ongelmitta toimivat testitunnukset. Tämän pitäisi käydä melko helposti, koska molemmissa tällä hetkellä keskeneräisissä testeissä jäljellä on enää vain muutama vaihe verkkopankissa tehtävän tunnistautumisen jälkeen.

Aloitin tänään uuden laajakaistan tilaavan testin automatisoinnin ja tämän testin työstämisestä tuli päivän uusi tavoite. Kyseisen testin automatisoinnin aloitin kuitenkin hieman eri lähtökohdasta kuin muiden verkkokaupan testien. Kyseisen tilausprosessin läpikäymisen oli vielä kesken, koska aikaisemmin käyttämälläni testidatalla sivusto antoi virheilmoituksen uutta laajakaistaa tilattaessa. Saatuani käyttöön lisää hyödynnettävää testidataa, pystyin samalla testiä automatisoidessani kirjoittamaan sen eri vaiheet ja niiden työstämisen etenemisen ylös Xmindiin. Samalla koko tilausprosessi tuli minulle entistä tutummaksi.

Pääsin päivän aikana hyvin etenemään laajakaistatilauksen automatisoinnissa ja sain omasta mielestäni saavutettua tälle päivälle asettamani tavoitteen. Päivän suurimmaksi haasteeksi nousivat nimeltään ja id-tunnisteeltaan saman nimiset web-elementit. Kyseessä oli siis tilanne, jossa käyttäjä valitsee hänen ilmoittamaansa osoitteeseen saatavilla olevista laajakaistapaketeista haluamansa paketin. Nämä eri laajakaistapaketit (tässä tapauksessa kolme eri pakettia) löytyivät verkkokaupan testisivustolta sekä nimeltään että rakenteeltaan identtisistä elementeistä. Koska elementit olivat saman nimisiä ja sisälsivät

myös lisää identtisiä elementtejä, Robotilla oli vaikeuksia painaa muun muassa ”Jatka tilausta”-painiketta. Tämä johtui siitä, että yhden painikkeen sijasta annettu XPath-polku viittasi kaikkien kolmen elementin painikkeeseen. Ongelma ratkesi sillä, että aktivoitaessa haluttu laajakaistapaketin sisältämä elementti, valitulle elementille ja sen alielementeille tuli näkyviin attribuutti, jota muilla kahdella elementillä ei ollut. Tämän attribuutin avulla XPath-polkuun pystyi yksilöimään mistä elementistä oli kyse ja Robotti pystyi löytämään halutun elementin muiden vaihtoehtojen joukosta.

Keskiviikko 5.7.2017

Päivän tavoitteena oli työstää eilen aloittamaani uuden laajakaistan tilaamisen suorittavaa testiä niin pitkälle kuin mahdollista puuttuvien testitunnuksia silmällä pitäen. Tämän jälkeen tarkoituksenani oli aloittaa seuraavan testin työstäminen. Sain nopeasti saavutettua päivän päätavoitteeni ja automatisoitua laajakaistan tilauksen verkkopankissa tehtävään tunnistautumiseen asti. Lukuun ottamatta valmiiksi saamaani matkapuhelimen tilausta suoramaksulla, kaikki verkkokauppaa koskevat testit olivat nyt automatisoitu niin pitkälle kuin mahdollista ilman toimivia testitunnuksia.

Seuraavaksi aloitin verkkokaupan testiautomaatioiden koodin hiomisen. Sain lyhennettyä sekä selkeytettyä koodia yhdistämällä ne vaiheet yhteisiksi testitapauksikseen, jotka ovat jokaisessa verkkokaupan testissä samat. Esimerkiksi verkkokaupan pääsivulta siirtyminen eri tuotekategorioiden sivuille oli kaikissa testeissä samanlainen testitapaus, jossa vain klikattavien linkkien nimet ja aukeavat sivut vaihtelivat tapauskohtaisesti. Toinen vastaava testitapaus oli ostoskorin tarkistaminen ennen tilauksen jatkamista vain ostoskorista löytyvien tuotteiden ja niiden hintojen vaihtuessa testistä riippuen.

Testitapausten yhdistäminen kuitenkin vaati sen, että yhdisteltävissä vaiheissa verkkokaupan käyttöliittymä pysyisi testikohtaisia muuttujia, kuten tuotteen nimeä, lukuun ottamatta samana. Tämä vaatimus oli sitä varten, jotta Robotti pääsisi aina testistä riippumattomiin elementteihin käsiksi. Tällainen koodin siivoaminen ja uudelleen jaksottaminen oli oppisen kannalta hyvin tärkeää, koska sen kautta opin uusia ja erilaisia tapoja rakentaa testitapauksia. Jokaiselle testille samoja testitapauksia yhdistämällä vältyin myös turhalta toistolta ja testien rakenteen hallinta helpottui samanaikaisesti.

Iltapäivästä sain vihdoin haltuuni uusia toimivia testikäyttäjätunnuksia ja pääsin palaamaan ja viimeistelemään keskeneräisiä verkkokaupan testejä. Otin ensin työn alle uuden mobiililiittymän tilauksen ja sain viimein kyseisen testin ohjelmoitua. Vaikka sain testin ohjelmoitua, se ei kuitenkaan ollut vielä täysin valmis, koska sain tehtäväkseni tehdä koo-

diin vielä lisää varmistuksia. Tällaisilla varmistuksilla esimerkiksi tarkistetaan, että tilattavan tuotteen hinta pysyy samana koko tilausputken läpi ja että verkkokaupan käyttöliittymään eri järjestelmistä haettavat asiakastiedot vastaavat Robottiin syötettyjä tietoa. Näitä tarkistuksia tehdessäni opin muun muassa vertaamaan verkkosivulta löytyvän elementin sisältämää arvo Robottiin syötettyyn muuttujan arvoon. Tarkastuksia tehdessäni lisäsin ne samalla Xmindin-tiedostoon työn dokumentoinnin parantamiseksi.

Sain saavutettua päivälle asettamani tavoitteet ja olin myös tyytyväinen siihen, miten oppimiseni Robot Frameworkiin liittyen kehittyi taas eteenpäin muun muassa testiyhteisiä testitapauksia luomalla ja erilaisia tarkistuksia ohjelmoimalla. Tarkoituksenani on huomenna jatkaa vielä uuden mobiililiittymän tilausprosessin työstämistä, koska siihen liittyviä tarkistuksia oli vielä tekemättä. Saadessani kyseisen testin täysin valmiiksi, pystyn viimeistelemään myös keskeneräiset testit, jotka käsittelevät uuden laajakaistan tilaamista sekä mobiililaitteen tilaamista osamaksulla.

Torstai 6.7.2017

Asetin eilen tämän päivän tavoitteeksi jatkaa mobiililiittymän tilausprosessiin liittyvien erinäisten tarkistusten tekemistä. Tavoitteenani oli pääsääntöisesti keskittyä tietyn kokoisen puhelinliittymän hintatietojen (normaalihinnan ja mahdollisen alennetun hinnan) talteen ottamiseen ja näiden hintatietojen vertailuun ostoskorissa oleviin hintoihin. Joudun ohjelmoimaan kyseisiä hintatietojen vertailuja, koska varsinainen hinta ja mahdollinen alennettu hinta haetaan eri järjestelmistä. Ohjelmoimissani verkkokaupan testiautomaatioissa olisi siis hyvä tarkistaa, että tuotteen hinta pysyy samana läpi tehtävän tilauksen.

Hintoja vertailtaessa Robotti noutaa tarvittavat hintatiedot sivustolta käyttäen ”Get Text”-avainsanaa, jonka avulla eri elementeistä voidaan noutaa niiden sisältämä tekstijono (Robot Framework Foundation, 2017d). Avainsanan avulla Robotti hakee ensin tuotesivulta tuotteen hinnat tekstimuodossa määritellyistä elementeistä ja tallentaa ne omiksi muuttujikseen. Siirryttäessä ostoskorisivustolle tilauksen eri vaiheissa, Robotti käy hakemassa ostoskorista löytyvät hintatiedot ja vertaa niitä aikaisemmin muuttujiksi tallennettuihin tietoihin. Jos hintatiedot täsmäävät, Robotti siirtyy tilauksessa eteenpäin. Tähän hyvänä mallina ja pohjana toimivat kollegani kokoamista esimerkeistä löytyvä testitapaus, jossa haetaan Robotin suorittaman testin suoritusvauhti ja tallennetaan se omaksi muuttujakseen.

Mobiililiittymän tilauksen automatisoinnin valmistumisen jälkeen siirtyessäni ohjelmoimaan muiden testien hintatarkistuksia, ongelmaksi nousivat tuotteiden eri maksuvaihtoehdot. Tilattaessa uutta laitetta tai liittymää kuukausimaksulla, ostoskorista löytyvät hintatiedot

olivat eri nimisissä elementeissä kuin kertamaksulla tilattaessa. Näin ollen joudun tekemään molemmille maksuvaihtoehdoille omat tarkistuksensa. Tämän ei pitäisi olla kuitenkaan liian työlästä, koska pystyn tässä tilanteessa hyödyntämään jo ohjelmoimiani ostoskorin hintatietoja tarkistavaa testitapausta muuttamalla tutkittavien elementtien nimet.

Hintatietojen tarkistuksen ohella tein niiden lisäksi myös ylimääräisen, mutta lopulta hyvin tarpeellisen, tarkistuksen tilausprosessin vaiheeseen, jossa Robotti syöttää käyttäjän tiedot Tupas-tunnistautumiseen vaadittaviin tekstikenttiin. Osa saamieni uusien testitunnuksien tiedoista oli määritelty muihin järjestelmiin niin, että testikäyttäjän sekä etu- että sukunimi löytyivät yhdestä muuttujasta kahden sijaan. Yleensä tiedot oli tallennettu vain sukunimikenttään, jolloin testitunnuksen etunimitiedot olivat jääneet tyhjiksi. Tämän vuoksi ohjelmoin Robot Frameworkiin ylimääräisen testitapauksen.

Testitapauksessa Robotti tarkisti, että jos olin määritellyt käsin muiden järjestelmien tietojen perusteella testitunnuksen joko etu- tai sukunimen arvon tyhjäksi, Robotti syöttää määrittelemästäni kolmannesta muuttujasta käyttäjän koko nimen tarvittavaan kenttään. Näin vältytään siltä, että koko testi epäonnistuisi sen takia, että jos testikäyttäjän nimitiedot olisivat puutteelliset. Vastaavasti taas lisäsin samaan tarkistukseen ehdon, että jos testikäyttäjältä löytyvät sekä etu- että sukunimitiedot, Robotti syöttää nimitiedot normaalisti vaadittaviin kenttiin.

Päivän aikana opin täysin uutena asiana hakemaan Robot Frameworkilla tutkittavasta elementistä teksti-tai numeroarvon. Hintatietoja haettaessa käsittelin löydettyä hinta-arvoa niin, että Robotti ottaa siitä vain talteen numeerisen arvon ja jättää muut elementistä löytyvät merkit pois eri merkkijonon pilkkomiseen tarkoitetuilla avainsanoilla. Pelkän numeroarvon talteen ottaminen helpottaa huomattavasti tehtävää hintavertailua, koska tällöin vertailua eivät sekoita välilyönnit tai muut erikoismerkit. Vastaavasti esimerkiksi tuotetietoja tai käyttäjän nimi- ja osoitetietoja vertailtaessa, muutin haetut tekstijonot isoilla kirjaimilla kirjoitetuksi, jotta arvojen vertailu onnistuu mahdollisimman vaivatta. Esimerkiksi jos verkkokaupan käyttöliittymässä ja Robotin muuttujassa käyttäjän nimitiedot on muotoiltu eri tavalla, tietojen vertailu onnistuu silti vaivatta.

Päivän suurimpana haasteena oli määritellä Robotin muuttujan arvo niin, että se kulkisi usealle riville jatkuvan Run Keyword If-avainsanan läpi. Run Keyword If-avainsanan avulla Robotti suorittaa ehtolauseessa annetun avainsanan vain, jos ehtolauseeseen määritelty ehto on tosi (Robot Framework Foundation, 2017a). Yritin kyseisen ehtolauseen avulla hakea tietyn elementin arvon ja asettaa sen muuttujaksi, vain jos aukeavan sivun otsikko vastaisi haluttua tekstijonoa. Nyt Robotti antoi vain virheilmoituksen, että ehtolauseessa

käsiteltävän muuttujan arvoa ei olisi määritelty ollenkaan. Yritin määrittää haettavan muuttujan arvoa ehtolauseen sisällä ja sitä ennen, mutta tuloksetta. Etsin Robot Frameworkin materiaaleista ja muista lähteistä ratkaisua tähän ongelmaan, mutta en saanut ongelmaa vielä tämän päivän aikana selvitettyä. Töiden etenemisen vuoksi jätin keskeneräisen ehtolauseen hautumaan ja toteutin tarvittavien tietojen hakemisen eri testitapauksella. Tavoitteenani on yrittää saada perjantaina selvitettyä tämä ongelma, jotta se ei tulisi hidastamaan muiden testiautomaatioiden ohjelmointia jatkoa silmällä pitäen.

Perjantai 7.7.2017

Aloittaessani eilen erilaisten tarkistusten tekemisen uuden mobiililiittymän tilaukseen, asetin samalla tämän päivän tavoitteeksi jatkaa niiden työstämistä. Tavoitteenani on saada kyseisen testin automatisointi valmiiksi kaikkine siihen liittyvine tarkistuksineen. Näillä tarkistuksilla tarkoitan Robot Frameworkilla tehtyjä testitapauksia, jotka ajetaan tietyissä kohdissa testiä ja joilla esimerkiksi tarkistetaan, että tilattavan tuotteen oikea hinta kulkee muuttumattomana koko tilausprosessin läpi.

Sain päivälle asettamani tavoitteet saavutettua, vaikka en saanutkaan kaikkia tarkistuksia tehtyä ja viimeistelyä mobiililiittymän tilausta suorittavaa testiä. Ohjelmoimatta jäi ainoastaan tuotteen hinnan ja muiden tietojen tarkistaminen tilausvahvistussivustolta, joka tulee näkyviin tilauksen onnistuneen lähettämisen jälkeen. Kun kyse on mobiililiittymän tilaamisesta, tarkistukseen kuuluu myös, että Robotti tarkistaa, että tilausvahvistuksesta löytyy uusi valittu puhelinnumero. Lisäsin myös eilen tekemääni hintatietojen tarkistamiseen ominaisuuden, jossa Robotti noutaa tuotteen kokonaishinnan, samalla tavalla kuin muutkin tuotteelle määritellyt hinnat, tallentaa sen muuttujaksi ja vertaa sitä toiseen haluttuun muuttujaan, tässä tapauksessa nykyiseen hintaan.

Päivän oppiminen tapahtui pääsääntöisesti kertauksen kautta työstäessäni jo aloittamiani ja uusia tarkistustestitapauksia. Suurin oppiminen ja kehitys tapahtuivat siinä, kun sain selvitettyä eilen vastaan tulleen ongelman liittyen muuttujan arvon määrittelyyn Run Keyword If-ehtolauseen sisällä. Tarkoituksena oli siis hakea muuttujalle arvo verkkokaupan käyttöliittymästä ainoastaan, jos ehtolauseen ehto tuli täyteen.

Etsittyäni koko eilisen ja tämän iltapäivän ratkaisua ongelmaan löysin siihen vihdoinkin vastauksen Stack Overflow-sivustolta, jossa toinen käyttäjä oli törmännyt samanlaiseen ongelmaan. Ratkaisuna muuttujan arvon määrittämiseen ehtolauseessa, oli määrittää se suoraan itse ehtolauseeseen, ei sen sisälle. Tätä ennen oli kannattavaa myös ennen ehtolausetta määrittää muuttujan arvoksi tyhjäksi tai nollaksi, jotta se on varmasti tyhjä en-

nen sen uudelleen määrittämistä (Stack Overflow, 2016). Vaikka sainkin muuttujan määriteltyä haluamallani tavalla, päädyin silti pysymään eilen tekemässäni ratkaisussani, koska se oli koodin rakennetta ja selkeyttä silmällä pitäen parempi ratkaisu. Silti oppimisen kannalta oli hyvin tärkeää saada eilen vastaan tullut ongelma ratkaistua, jotta se ei enää tuotaisi ongelmia tulevaisuudessa.

3.4.1 Viikkoanalyysi 4

Oma osaamiseni kehittyi neljännen seurantaviikon aikana taas eteenpäin niin uusia asioita oppimalla kuin vanhoja kertaamalla. Pääasiassa tämän viikon oppiminen tapahtui aloitettujen testien automatisointia jatkamalla lisäämällä niihin uusia tarvittavia testitapauksia, kuten erilaisia varmistuksia. Uusia testitapauksia ohjelmoidessani olen huomannut, että osaamiseni karttuessa olen pystynyt kokeilemaan erilaisia ja uusia tapoja hyödyntää jo minulle entuudestaan tuttuja Robot Framework-avainsanoja ja kirjastoja. Löytämällä uusia toimintatapoja pystyn mahdollisesti tekemään entistä selkeämpää ja paremmin jaksotettua koodia.

Tämän viikon suurimpana yksittäisenä ongelmana olivat edelleen testitunnukset ja niiden puuttuminen. Tämä sama ongelma oli läsnä jo viime viikolla. Ongelmat liittyivät pääsääntöisesti testitunnuksille asetettuun rajaan siitä, kuinka monta kertaa yhdellä tunnuksella pystyy suorittamaan verkkopankissa tehtävän sähköisen tunnistautumisen. Toinen testitunnuksiin liittyvä ongelma oli se, että käyttämälläni testitunnuksella ei myöskään ollut tarvittavat luottotiedot kunnossa tietyn tilausprosessin loppuun suorittamiseksi. Kunnollisten testitunnuksien puuttumisen takia en pystynyt viime viikolla enkä tämän viikon alkupuolella viemään tiettyjä aloittamiani testejä loppuun. Tästä johtuen minulla oli alkuviikosta käsissäni useita keskeneräisiä verkkokaupan tilausprosesseja käsitteleviä testejä.

Testitunnuksiin liittyvät ongelmat kuitenkin ratkesivat viimein keskiviikkona kun sain eräältä kollegaltani useamman uuden testitunnuksen käyttööni aikaisemman yhden sijasta. Koska sain pelkästään testikäyttäjän testihenkilöturvatunnuksen, minun oli itse etsittävä niiden järjestelmien testi-instansseista, jotka säilyttävät asiakasdataa, muut kyseisten testikäyttäjien tiedot. Tämä sujui ongelmitta tarvittaviin järjestelmiin liittyvän tietoperustani ollessa jo melko hyvä. Saatuani lisää testitunnuksia pääsin vihdoin jatkamaan automaatioiden ohjelmoimista. Ennen kuin voin kuitenkin viimeistellä testin uuden mobiililaitteen tilaamiseksi rahoituksella, minun on ensin tarkistettava millä uusista testikäyttäjätunnuksista on tarvittavat luottotiedot kyseisen tilausprosessin loppuun viemiseksi.

Toinen viikon suurimmista ongelmista liittyi jo työstämiini testeihin lisättäviin uusin testitapauksiin. Näiden testitapauksien, tarkoituksena oli suorittaa erilaisia tarkistuksia, jotka varmistavat, että tilattavan tuotteen tiedot, kuten nimi ja hintatiedot, pysyvät samana koko tilausputken läpi. Pääasiallisesti näiden uusien testitapausten haasteena oli löytää oikeat tavat ja avainsanat niiden toteuttamiseksi, joka vaati myös uusiin Robot Framework-kirjastoihin, kuten String-kirjastoon, tutustumista. String-kirjasto sisältää erilaisten tekstijonojen käsittelyyn ja muotoiluun tarkoitettuja avainsanoja (Robot Framework Foundation, 2017e). Uusien toimintatapojen, avainsanojen ja kirjastojen käyttöönotto vaati olemassa olevan Robot Framework-materiaalin tutkimista entistä laajemmin sekä paljon kokeilua ja virheiden kautta oppimista. Ongelmien selvittämisessä auttoivat paljon myös erilaiset foorumisivustot, jonne käyttäjät ovat voineet esittää kysymyksiä Robot Frameworkiin ja sen käyttöön liittyen ja näihin kysymyksiin muut käyttäjät ovat voineet vastata. Tällaisia sivustoja ovat olleet muun muassa Stack Overflow- ja Google Groups-sivustot.

Stack Overflown ja Google Groupsin kaltaisia sivustoja tutkimalla pystyin hyvin tutustumaan muiden Robot Frameworkin käyttäjien toimintatapoja ja -malleja koodin ohjelmoimisessa ja sen järjestelyssä. Näitä muiden käyttämiä toimintatapoja tutkimalla pystyin hyvin tarkastelemaan omia ohjelmointi- ja toimintatapojani uudesta, kriittisemmästä näkökulmasta. Näin sain itseni pohtimaan aktiivisesti miten pystyisin kehittämään omaa toimintaani. Koska ongelmatilanteissa apunani on sekä internetistä että tiimimme versionhallinnasta löytyvät esimerkit, pystyn jatkuvasti vertailemaan omaa toimintaani niihin, etsimään uusia menetelmiä ja tapoja erilaisten toimintojen toteuttamiseksi.

Uusien toimintatapojen ja menetelmien etsimisestä on ollut hyötyä varsinkin nyt, kun Robot Frameworkin oppimiseni on kehittynyt kuluneen viiden viikon aikana kehittynyt jatkuvasti eteenpäin ja pystyn hyödyntämään myös materiaaleja, joita vasta aloitteleva Robotin käyttäjä ei pysty välttämättä vielä hyödyntämään. Näin pystyn laajentamaan omaa osaamistaan entuudestaan ja samalla löytämään itselleni sopivammat toimintatavat Robot Frameworkin ohjelmoimiseen.

3.5 Seurantaviikko 5

Maanantai 10.7.2017

Tämän päivän tavoitteena oli viimeistellä uuden mobiililiittymän tilausprosessin automatisointi erilaisine tarkistuksineen. Testin viimeistelyyn ei pitäisi kulua hirveästi aikaa, koska jäljellä oli vain itse käyttäjän ja tuotteen tietojen tarkistaminen tilausvahvistussivulta. Sain

saavutettua päivälle asettamani tavoitteen ja viimeisteltyä kyseisen tilausprosessin automatisoinnin.

Puuttuvia testitapauksia ohjelmoidessani pystyin taas hyödyntämään aikaisemmin ohjelmoimiani tarkistuksia tekeviä testitapauksia, joten minun ei tarvinnut ohjelmoida kaikki puuttuvia tarkistuksia alusta asti. Esimerkiksi tuotteen hintatietojen tarkistukseen tilausvahvistuksessa pystyin hyödyntämään muuttamattomana testitapausta, joka tarkistaa tilauksen aikaisemmissa vaiheissa, että ostoskorista löytyvät hintatiedot ovat oikein. Tämä johtui siitä, että ostoskorista ja tilausvahvistuksesta löytyvät tuotteen hintatiedot sisältävät elementit olivat molemmissa tapauksissa saman nimisiä. Esimerkiksi asiakkaan tietojen ja uuden valitun puhelinnumeron tarkistamisessa en pystynyt suoraan hyödyntämään valmiita testitapauksia, joten näitä tilanteita varten jouduin luomaan uusia testitapauksia. Pystyin kuitenkin myös näissäkin tapauksissa hyödyntämään jo valmiita testitapauksia malleina ja pohjina.

Opin päivän aikana taas paljon uutta Robot Frameworkista muun muassa opettelemalla käyttämään uusia avainsanoja ja tarkastelemalla tekemääni koodia entistä kriittisemmästä näkökulmasta sen laadun parantamiseksi. Uusia avainsanoja pääsin hyödyntämään esimerkiksi Robotilla haettua tekstijonoa pilkottaessa. Tekstijonon pilkkomisessa käytin Fetch From Right- ja Fetch From Left-avainsanoja, joissa Robotti hakee avainsanan nimen mukaisesti halutusta tekstijonosta määritellyn mittaisen tekstijonon halutusta suunnasta (Robot Framework Foundation, 2017e). Tekemäni koodin rakennetta sain parannettua taas entuudestaan jakamalla esimerkiksi muutamia hieman liian pitkiä testitapauksia useammiksi, omiksi testitapauksikseen**.** Uusimalla muutamien eri testitapauksien rakenteita, koodista tuli huomattavasti selkeänpää ja helpommin luettavaa.

Tiistai 11.7.2017

Koska sain eilen vietyä loppuun uuden mobiililiittymän tilausprosessin automatisoinnin, seuraavana tavoitteena oli jo aloittamani uuden laajakaistan tilaamisen automatisoinnin jatkaminen. Kyseisen tilausprosessin viimeiset vaiheet olivat myös vielä kartoittamatta aikaisemmin puutuneista testitunnuksista johtuen.

Päivän oppiminen tapahtui pääsääntöisesti kertauksen kautta, ohjelmoimieni testitapausten rakenteiden ollessa hyvin selkeitä jo aikaisemmin tällä viikolla valmiiksi saamani testin takia. Kertaus ei tietenkään ole koskaan pahasta, vaikka se saattaa välillä tuntuakin turhauttavalta toistolta, mutta ainakin itseni kohdalla kertauksen kautta niin teoria kuin käytäntökin jäävät paremmin mieleen. Tämän jälkeen omasta toiminnasta tulee huomattavas-

ti luonnollisempaa ja myös oman osaamisen tarkastelu ja arviointi helpottuu. Vaikka uuden laajakaistan tilausprosessi olikin sen loppuvaiheiden osalta kartoittamatta, en varsinaisesti oppinut uutta varsinaiseen tilausprosessiin liittyen, koska ne vaiheet, jotka olivat vielä läpikäymättä, olivat täysin samat kuin osa mobiililiittymän tilauksen viimeisimmät vaiheet.

Laajakaistan tilausprosessia työstäessäni huomasin, että käytännöllisyyden ja oppimisen kannalta pyrin ohjelmoimaan yhteisiä, mahdollisimman monessa tilausprosessissa hyödynnettäviä testitapauksia. Tämä ei kuitenkaan ollut aina helpoin ratkaisu mihin vaikutti muun muassa se, että kaikki tilausprosessit eivät olleet identtisiä ja tilauksista riippuvaiset muuttujat ja web-elementit aiheuttivat myös ongelmia. Käytännöllisyydestään huolimatta kaikkia testitapauksia ei ole kuitenkaan mahdollista rakentaa kaikkiin testeihin sopiviksi ja yhteisiä testitapauksia ohjelmoidessa on myös hyvä olla pelisilmää ja toimia tilanteen mukaan. Ei ole kannattavaa yrittää ohjelmoida monissa eri testeissä toimivia testitapauksia, jos esimerkiksi testitapauksen koodista tulee tästä syystä liian epäjohdonmukaista ja sekavaa. Mielestäni aina on pyrittävä testien ja testitapausten rakennetta sekä yleisesti Robot Framework-koodin selkeyttä ja yhdenmukaisuutta silmällä pitäen parhaaseen ratkaisuun.

Työskennellessäni pyrin jatkuvasti tutkimaan koodiani ja pohtimaan miten sitä pystyisi selkeyttämään ja hiomaan esimerkiksi yhteisten muuttujien ja testitapauksien avulla. Samalla huomasin tehneeni aikaisemmin ratkaisuja, jotka näin jälkeenpäin vaikuttivat epäkäytännöllisiltä. Tämä johtui siitä, että aikaisemmin minulla ei ollut vielä samaa tietoperustaa ja käytännön kokemusta Robot Frameworkista kuin mitä minulla nyt on. Olin näihin omiin huomioihini tyytyväinen, koska koodin rakenteen tutkiminen ja kritisointi sekä koodin laadun parantaminen ovat mielestäni osa Robot Frameworkiin liittyvän oppimisen seuraavia vaiheitta. Tutkiessani omaa toimintaani jatkuvasti entistä kriittisemmästä näkökulmasta olen oppinut paljon uutta ja olen löytänyt myös uusia toimintatapoja ja -malleja, joita en olisi neljä viikkoa sitten tullut ajatelleeksikaan.

Keskiviikko 12.7.2017

Päivän tavoitteenani oli viimeistellä eilen työstämääni laajakaistan tilaamisen automatisointia. Päivälle asettamani tavoite oli kuitenkin hyvin saavuttavissa. Tämä johtui siitä, että minun oli vain tarkistettava pystyinkö hyödyntämään tilausprosessin viimeisissä vaiheissa samoja koodeja kuin mobiililiittymän tilauksessa vai oliko minun ohjelmoitava uusia testitapauksia vanhojen testitapauksien pohjalta.

Sain tälle päivälle asettamani tavoitteet saavutettua saadessani viimeisteltyä laajakaistan tilaamisen automatisoinnin. Tilausprosessin viimeisten vaiheiden ohjelmoimisessa, kuten käyttäjän tietojen tarkistamisessa ja tilauksen vahvistamisessa, pystyin kuin pystyinkin hyödyntämään mobiililiittymän tilauksen viimeisissä vaiheissa käyttämiäni testitapauksia lähes muuttamattomana. Tilausvahvistuksen tarkistamisen jouduin kuitenkin tekemään omana testitapauksenaan, koska laajakaistaa tilattaessa tilausvahvistus sisälsi enemmän elementtejä kuin mobiililaajakaistaa tilattaessa. Saatuani loputkin verkkokaupan tilausprosessien automaatiot valmiiksi, pyrin siistimään ohjelmoimani koodin niin, että päällekkäiset ja saman asian tekevät testitapaukset saataisiin karsittua pois.

Saatuani tämän päivän aikana valmiiksi laajakaistan tilausprosessin automatisoinnin puolet verkkokaupan testeistä olivat valmiit. Kahdessa jäljellä olevassa tilausprosessissa ei ollut loppujen lopuksi hirveästi tehtävää, koska toisesta näistä testeistä puuttui vain tarvittavat tarkistukset ja toisesta tarkistusten lisäksi vain muutama tilausprosessin viimeisimmistä vaiheista.

Torstai 13.7.2017

Saatuani eilen valmiiksi uuden laajakaistan tilaamisen automatisoinnin, jäljellä olivat enää testit, joissa uusi mobiililaite tilataan suora- ja osamaksulla. Päivän tavoitteena oli näiden kahden keskeneräisen testin viimeistelyn lisäksi valmiin koodin hiominen tarpeen vaatiessa.

Ensimmäiseksi aloitin erilaisten tarvittavien tarkistuksien lisäämisen näihin mobiililaitteen tilauta käsitteleviin testeihin. Kuten muissakin verkkokaupan tilausprosesseissa, pystyin hyödyntämään näissä testeissä muissa testeissä käytettyä koodia muuttamalla ainoastaan testikohtaisten muuttujien ja web-elementtiä arvoja. Koska näiden kahden työstettävän tilausprosessin vaiheet olivat ennen verkkopankkiin liittyviä vaiheita täysin samat pystyin työstämään molempia tilausprosesseja samanaikaisesti. Näin pystyin kerralla työstämään molemmissa prosesseissa esiintyvät vaiheet ja testitapaukset. Tämän jälkeen pysytyin suoraan siirtymään niihin vaiheisiin, jotka olivat molemmissa tilausprosesseissa yksilöllisiä.

Saavutin päivän tavoitteeni saatuani työstettyä molempia mobiililaitteen tilauksen automaatioita eteenpäin lisäämällä niihin tarvittavia tietojen tarkistusvaiheita. En kuitenkaan saanut tänään kaikkia testien vaiheita valmiiksi. Molemmista prosesseista puuttuivat enää molemmille yksilölliset, tilauksien loppuosia koskevat vaiheet. Näitä testejä työstäessäni sain samalla taas parannettua koodini rakennetta. Päätin, että saatuani kaikki verkko-

kauppaan liittyvät automaatiot valmiiksi, yrittäisin tutkia ennen seuraavien järjestelmien testaamiseen siirtymistä pystyisinkö saamaan mobiililaitteen tilaukseen liittyviä testejä varten ohjelmoimani, ostoskorin tietoja tarkistavan testitapauksen toimimaan kaikissa verkkokaupan testeissä.

Perjantai 14.7.2017

Tämän raportointipäivän tavoitteena oli saada valmiiksi eilen työstämäni testit uuden mobiililaitteen tilaamiseksi suora- ja kuukausimaksulla. Tämän jälkeen tavoitteena oli keskittyä verkkokaupan testiautomaatioiden koodin siistimiseen ja pyrkiä saamaan eilen ohjelmoimani uusi testitapaus ostoskorin sisältävien tietojen tarkistukseen liittyen toimimaan jokaisessa verkkokaupan testissä.

Sain päivän aikana valmiiksi mobiililaitteen tilaamisen automatisoinnin suoramaksulla, mutta laitteen tilaamista kuukausimaksulla en kuitenkaan saanut valmiiksi. Tämä johtui siitä, että käyttämässäni testiympäristössä tai – sivustossa ilmeni teknisiä ongelmia, joiden vuoksi tilausprosessin loppuun vieminen ei ollut mahdollista edes manuaalisesti. Ongelmat ilmenivät tilausprosessissa sähköisen tunnistautumisen jälkeen, kun sivusto ilmoitti, että palvelussa on tällä hetkellä virhe. Tämän vuoksi tilausprosessin loppuun viemistä oli yritettävä myöhemmin. Kyseinen ongelma ei tämän päivän aikana vielä ratkennut ja huonolla onnella ongelman ratkeamiseen menee tavallista kauemmin johtuen muiden työntekijöiden pitkistä lomista näin heinäkuussa.

Teknisten ongelmien takia siirryin viimeistelemään ja hiomaan muita valmiita verkkokaupan testejä. Tilausprosessien viimeistelemiseksi kävin ne yksi kerrallaan läpi ja yhdistin useita testitapauksia niin, että niitä voidaan hyödyntää joko kaikissa tai vain tietyissä verkkokaupan testeissä vain muuttamalla testitapaukseen liittyvien muuttujien arvoja ajettavasta testistä riippuen. Esimerkiksi tuotesivua avattaessa avattavan linkin nimi on määritelty muuttujaksi, jonka arvo riippuu tilattavasta tuotteesta. Muita ohjelmoimiani globaaleja testitapauksia olivat muun muassa käyttäjän tunnistautuminen verkkopankin kautta, tilauksen vahvistaminen ja ostoskorin tietojen tarkistaminen.

Päivän toisen tavoitteen eli eilen ohjelmoimani ostoskorin sisältämät tiedot tarkistavan testitapauksen saaminen toimimaan muissakin verkkokaupan testeissä osoittautui hyvin haastavaksi. Ongelmaksi nousivat erilaiset hintatiedot, joita saattoi tuotteesta riippuen olla enemmän kuin vain tuotteen kerta- ja kuukausihinta. Tällaisia ylimääräisiä hintatietoja olivat erilaisista tarjouksista johtuva tuotteen alennettu hinta tai tuotteen mukana tulevan lisälaitteen, kuten modeemin, hinta. Ongelman ratkaisemiseksi määritin ostoskorin tarkis-

tukseen jokaiselle eri hintatyypille omat muuttujansa, jotka määritellään testikohtaisesti. Ennen hintojen tarkistamista, Robotti tarkistaa ensin mitkä näistä muuttujista ovat näkyvissä ja tarkistaa hintatiedot näkyvien hintatietojen perusteella.

3.5.1 Viikkoanalyysi 5

Tämän viikon yhtenevänä teemana on ollut oman koodin tarkasteleminen kriittisestä näkökulmasta sen jaksottamiseksi, ryhmittelemiseksi ja hiomiseksi. Viimeistellessäni tämän ja viime viikon aikana verkkokaupan eri testejä, huomasin tarkastelevani omaa Robot Framework koodiani huomattavasti kriittisemmästä näkökulmasta kuin aikaisemmin. Kuten viime viikon viikkoanalyysissa mainitsinkin, olen nyt pystynyt rakentamaan koodiani täysin uudella tavalla hyödyntäen sekä uusia että minulle jo entuudestaankin tuttuja avainsanoja ja kirjastoja. Ohjelmoimalla esimerkiksi lisää testiyhteisiä muuttujia olen saanut poistettua päällekkäisiä testitapauksia ja tämän ansiosta koodistani tuli entistä selkeälukuisempaa ja yhdenmukaisempaa.

Ohjelmoinnin pääpiste ei ole enää ollutkaan pelkästään siinä, että tekemäni Robot Framework-testit toimisivat haluamallani tavalla vaan myös siinä, että tekemäni koodi olisi mahdollisimman hyvin kirjoitettua. Olen edesauttanut tätä hyödyntämällä tilanteisiin parhaiten sopivia avainsanoja ja kirjastoja, enkä ole vain yrittänyt saada tuttuja ja turvallisia menetelmiä toimimaan joka tilanteessa. Olen myös pyrkinyt välttämään koodin kirjoittamista niin, että turhat päällekkäiset testitapaukset jäisivät turhan toiston välttämiseksi pois. Pyrin myös kirjoittamaan mahdollisimman yksinkertaista koodia, jotta sitä on helppo seurata ja ylläpitää. Näillä keinoilla olen pystynyt jatkuvasti haastamaan itseäni ja etsimään entistä parempia sekä välillä myös helpompia tapoja kirjoittaa koodia.

Lisätessäni sekä valmiisiin, että vielä keskeneräisiin testeihin tarvittavia tilaustietojen tarkistusvaiheita, huomasin, kuinka tärkeä suunnitteluvaihe on eri projekteissa. Suunnitteluvaiheen tärkeyttä on painotettu varsinkin koulussa eri kursseilla ja sitä painotettiin myös aikaisemmassa työharjoittelussani Kaupunginkansliassa. Tämän huomasin työstäessäni verkkokaupan testejä, kun minulle ei heti informoitu tarpeeksi tarkkaan mitä ominaisuuksia testiautomaatioihin halutaan. Näin ollen jouduin jälkikäteen palamaan ja lisäämään niihin puuttuvia testitapauksia. Molempien osapuolien, sekä minun että kollegoideni, olisi tullut olla aktiivisemmin vuorovaikutuksessa toistemme kanssa, jotta myös tällaiset tarkemmat yksityiskohdat olisi saatu sovittua ennen töiden aloittamista. Nyt olin saanut jo yhden tilausprosessin kokonaan automatisoitua ja seuraavan työstettyä jo yli puolivälin.

Työskennellessäni minulle itselleni heräsi kysymys liittyen erilisten tietojen tarkistamiseen, ja päätin kysyä niistä eräältä kollegaltani. Tällöin kävi ilmi, että tällaiset ominaisuudet olisi hyvä lisätä testeihin. Oli hyvä, että asia selvisi, jotta tällaisten testitapausten tarvetta ei olisi painotettu vasta projektin loppumisen jälkeen. Keskenkin projektia oli jo työlästä lähteä lisäämään uusia testitapauksia keskelle valmista koodia ja tältä olisi ehkä pystytty vältymään, jos kommunikointi olisi ollut alussa entistä aktiivisempaa.

Viikon suurimmat ongelmat liittyivät loppuviikosta ilmenneisiin teknisiin ongelmiin, joista johtuen en saanut tällä viikolla valmiiksi verkkokaupan testejä, kuten olin viikon alussa suunnitellut. Kuten aikaisemmillakin viikoilla, haastetta tähänkin viikkoon toivat mukanaan web-elementtien XPath-polkujen määritteleminen. XPathin käyttöön ja erilaisten polkujen määrittämiseen liittyen olen tämän viikon aikana kuitenkin sisäistänyt entistä paremmin uusia toimintatapoja, joista on ollut paljon hyötyä web-elementtejä haettaessa ja joihin tutustuin W3Schoolsin-sivuilla.

Yksi näistä on ollut halutun elementin vanhempi- ja tytär- elementtien etsiminen hyödyntämällä /- ja //-merkkejä XPath-polussa. Kirjoitettaessa XPath-polkua, /-merkillä pystytään hakemaan polun vasemmalle puolelle määritellyn elementin tytärelementti eli kyseisen elementin ”alta” löytyvä seuraava elementti html-koodissa. //-merkinnällä taas pystytään hakemaan elementti, joka löytyy jostain aikaisemmin määritellyn elementin alta, jolloin kyseessä ei välttämättä ole tytärelementti (W3Schools, 2017b). Näiden merkintöjen käyttö osoittautui hyvin hyödylliseksi, koska niiden avulla pystyin rakentamaan tarvittaessa melko monimutkaisiakin XPath-polkuja.

Myös [n]-merkinnästä on ollut suuri hyöty XPath-polkuja määrittäessä. Sen avulla pystytään etsimään tietty elementti sen järjestysnumeron avulla, jota kirjain n kuvastaa. Esimerkiksi jos on tarve etsiä tietystä kohtaa html-koodia viides div-elementti, se merkattaisiin XPath-polkuun div[5]. Elementtien etsiminen järjestysnumeroa hyödyntämällä on hyvin käytännöllinen toimintatapa elementtien paikantamiseen varsinkin, jos kyseessä on useita samannimisiä ja – laisia elementtejä, joita ei pysty muulla tavoin yksilöimään. Järjestysnumerointia hyödynnettäessä on kuitenkin oltava tarkkana siinä, että verkkosivuja päivitettäessä, sivujen rakenne voi muuttua. Tällöin web-elementtien järjestysnumerot voivat muuttua ja aikaisemmin tietyllä järjestysnumerolla löytynyt elementti ei enää löydykään samalla järjestysnumerolla. Tästä huolimatta myös tämä XPathiin-liittyvä toimintatapa oli hyvin hyödyllinen.

3.6 Seurantaviikko 6

Maanantai 17.7.2017

Koska kaikki verkkokaupan testit olivat mobiililaitteen tilaamista kuukausimaksulla lukuun ottamatta valmiita, eivätkä käytettävään testiympäristöön liittyvät tekniset ongelmat olleet viikonlopun aikana verkkokaupan osalta ratkenneet, tämän viikon tavoitteena on aloittaa myyntijärjestelmän testien automatisointi. Tämän päivän tavoitteena oli aloittaa ensimmäisen myyntijärjestelmän testin ohjelmointi. Vaikka kyseessä on eri järjestelmä, web-käyttöliittymää testattaessa myös verkkokaupan testeissä käyttämäni toimintatavat ja ohjelmoimani koodit ovat hyödynnettävissä myös myyntijärjestelmää testattaessa.

Myyntijärjestelmän testikokonaisuuteen kuuluivat laajakaistan tilaaminen kuitu- ja kaapelitalouksiin lisäpalveluineen, netti-TV:n tilaus sekä F-Securen VPN-palvelun tilaus. Vaikka tämän järjestelmän kautta suoritetaan muutamien samojen tuotteiden tilauksia kuin verkkokaupankin kautta, kyseessä ovat silti eri tilausprosessit. Verkkokaupan tapauksessa kyseessä oli asiakkaan käyttöliittymä kun taas myyntijärjestelmässä kyseessä on yrityksen työntekijän, esimerkiksi kauppiaan, käyttöliittymä. Myyntijärjestelmän tapauksessa kyseessä ei ole uusmyynnistä vaan tilaukset tehdään yrityksen jo olemassa olevalle asiakkaalle.

Saavutin päivälle asettamani tavoitteen aloittamalla F-Securen VPN:n palvelun tilausprosessin automatisoinnin. Uudessa järjestelmässä tehtävien automatisointien aloittamisessa auttoi kollegani valmiiksi tekemä myyntijärjestelmän testi, jota pystyn käyttämään myös kaikkien myyntijärjestelmän testien pohjana. Valmiin myyntijärjestelmän esimerkkitestin ja ohjelmoimieni testien pitäisi sekä helpottaa että nopeuttaa uusien myyntijärjestelmien testien ohjelmoimista. Valmista testiesimerkkiä tutkiessani minulla oli kuitenkin aluksi hieman hankaluuksia täysin ymmärtää, mitä missäkin testitapauksessa tapahtui, koska koodi ei ollut minun itseni tekemää. Tähän vaikutti se, että kaikilla on oma tapansa kirjoittaa koodia, joka on vakiintunut eri toimintamalleja tutkimalla ja kokeilemalla. Kollegani taso Robot Frameworkin osaamiseen liittyen on myös hyvin paljon korkeampi kuin omani, joten koodista löytyi myös edistyneemmille käyttäjille tarkoitettuja avainsanoja ja toimintamalleja.

Tiistai 18.7.2017

Aloitettuani eilen ensimmäisen myyntijärjestelmän testiautomaation ohjelmoimisen, tämän päivän tavoitteena oli jatkaa VPN-palvelun tilausprosessin automaation työstämistä siitä

mihin eilen jäin. Asetin päivän tavoitteeksi saada kyseinen testi valmiiksi, jotta pääisisin siirtymään seuraavien myyntijärjestelmien testien pariin.

VPN:n tilausprosessin automatisoinnin jatkaminen sujui aluksi ongelmitta ja päivän työrytmillä sen olisi hyvin voinut saadakin valmiiksi. Päivälle asettamani tavoite ei kuitenkaan toteutunut johtuen tilausprosessin lopussa vastaan tulleesta ongelmasta. Varsinaisen tilausprosessin viimeiseltä sivulta löytyy web-sivuun upotettuna pdf-tiedostona sopimus tilattavasta tuotteesta ja/tai palvelusta. Sopimuksesta löytyy asiakkaan ja tilauksen tarkemmat tiedot.

Ongelmaksi nousi kyseisestä sopimuksesta löytyvien tilaus- ja asiakastietojen vertaaminen Robotin muuttujiksi tilauksen aikaisemmassa vaiheessa tallennettuihin tietoihin. Koska kyseessä on pdf-tiedosto, en onnistunut Robotilla hakemaan sopimuksesta tarkistettavia tietoja. En myöskään iltapäivän aikana löytänyt tietoa siitä, että onko Robotille edes olemassa erillistä kirjastoa, jonka avulla web-sivuun upotetun pdf-tiedoston sisältöä pystyisi tarkastelemaan.

Koska tilaukseen liittyvän sopimuksen tarkistaminen on viimeinen testitapaus, joka minun on enää ohjelmoitava sen jälkeisten vaiheiden ollessa jo valmiiksi kirjoitettuja kollegani toimesta, päätin, että jätän sopimuksen tarkistamisen hautumaan ja siirryn työstämään muita myyntijärjestelmän testejä. Päädyin tähän ratkaisuun sen takia, koska sopimuksen tarkistaminen kuuluu kaikkiin käsittelemiini myyntijärjestelmän tilausprosesseihin, joten ajattelin, että palaan siihen muiden myyntijärjestelmän testien ollessa samassa vaiheessa. Ajanhallinnallisesta näkökulmasta tämä on hyvä ratkaisu, koska tällöin kaikkien myyntijärjestelmän testien ollessa samassa vaiheessa, voin rauhassa ja kerralla keskittyä sopimuksen tarkasteluun liittyvään ongelmaan.

Keskiviikko 19.7.2017

Päivän tavoitteeni oli aloittaa seuraavien myyntijärjestelmän testien ohjelmoiminen ja palata eilen ongelmia aiheuttaneeseen, sivustoon upotetun pdf-tiedoston, tutkimiseen myöhemmin. Päätin aloittaa Telia TV:n tilausprosessin automatisoinnin ja kyseinen tilausprosessi toi heti mukanaan uusia haasteita.

Haastetta aiheutti esimerkiksi eräs alasvetovalikko, josta käyttäjä valitsee osoitteen, johon haluttu palvelu tilataan. Olin muissa tekemissä testeissäni onnistunut hakemaan alasvetovalikoista haluamiani arvoja, mutta nyt jostain syystä Robotti ei aluksi meinannut saada etsittyä arvoa valituksi. Robotti löysi halutun alkion valikosta ja sai klikattua sitä, mutta

alkio ei jostain syystä jäänyt aktiiviseksi. Kokeilin ongelmaan erilaisia toimintatapoja ja avainsanoja, mutta alkio ei edelleenkään jäänyt aktiiviseksi. Lopulta ongelma ratkesi ja vika ei ollutkaan käyttämissäni avainsanoissa. Ongelma liittyi omaan koodiini. Alun perin Robotin avasi ensin listan ja valitsi halutun alkion listasta. Kun muutin testitapausta niin, että Robotti etsii suoraan halutun alkion listasta avaamatta listaa ensin, testitapaus toimi ja Robotti sai oikean arvon haettua. Tätä valmista testitapausta voi myös hyödyntää tulevissa laajakaistan tilausprosesseissa, jos valikoiden alkioiden kanssa ilmenee samanlaisia ongelmia.

Vaikka sainkin aloitettua ja työstettyä uutta tilausprosessia, päivälle asettamani tavoite ei täysin toteutunut. Tämä johtui uusista teknisistä ongelmista, joiden takia en päässyt jatkamaan TV-palvelun tilausprosessin automatisointia loppuun asti. Syystä, jota en tiedä käyttämäni testisivusto ei näyttänyt valitsemalleni osoitteelle kaikkia saatavilla olevia tuotteita. Testasin asiaa useilla eri osoitteilla ja testitunnuksilla, mutta tuotteita, joiden pitäisi olla saatavilla käyttämissäni osoitteissa, ei näkynyt sivustolla. Lähdin selvittämään asiaa muilta myyntijärjestelmän testauksesta vastaavilta henkilöiltä ja heidän mukaansa kohtaamani ongelma saattoi johtua siitä, että kyseinen testiympäristö on muuten hyvin vähän käytössä.

Testiympäristön vähäisestä käytöstä ja näin ollen myös sen päivitysten puutteesta johtuen, voi olla, että testiympäristöstä löytyi virheellisiä kytköksiä tai konfiguraatioita muihin järjestelmiin, joista eri tuotteiden tietoja haetaan. Tästä todennäköisesti johtuen testisivusto ei pystynyt näyttämään oikeita tuotteita. Ilmoitin ongelmasta eteenpäin ja siirryin työstämään seuraavia myyntijärjestelmän tilausprosesseja, mutta niissä ilmeni samana ongelma eikä annettuihin osoitteisiin löytynyt tuotteita, joiden pitäisi olla niihin saatavilla.

Torstai 20.7.2017

Tämän päivän tavoitteet ja niiden asettaminen riippuivat siitä, olivatko käyttämäni testiympäristön ongelmat ratkenneet päivän aikana vai eivät. Jos eilen vastaan tulleet testiympäristön ongelmat eivät olisi ratkenneet, olisin asettanut tavoitteeksi hioa jo valmista koodiani entuudestaan ja miettinyt mitä tilausprosessia pystyisin automatisoimaan seuraavaksi.

Koska testiympäristöön liittyvät kriittisimmät ongelmat oli saatu korjattua, pystyin jatkamaan laajakaistan tilausprosessin automatisointia ja päivän tavoitteeksi muodostui kyseisen testin vieminen eteenpäin. Saavutin tämän päivän tavoitteen ja sain laajakaistan tilausprosessin automatisoitua samaan vaiheeseen kuin missä VPN-tilaus oli eli jäljelle oli vain sivuun upotetun PDF-tiedoston tutkiminen. Saavutettuani päivän tavoitteen siirryin

työstämään tilausprosessia, jossa tilattava laajakaista on kuitutalouden sijasta kaapelitalouteen toimitettava liittymä. Tässä prosessissa pystyin käyttämään lähes muuttumattomana kuituyhteyden tilaukseen tekemiäni testitapauksia mikä nopeutti huomattavasti uuden testin ohjelmointia tilausprosessien ollessa muutamia eri vaiheita lukuun ottamatta täysin samat.

Päivän aikana oppimista tapahtui työstäessäni jo valmiita ja vielä keskeneräisiä myyntijärjestelmän testejä. Näitä testejä ohjelmoidessani en ollut aluksi ottanut huomioon, että eri tuotteiden ja palveluiden tuotesivut sisältäisivät samasta käyttöliittymästä huolimatta eri lukumäärän elementtejä, joten tuotesivuilta löytyvien elementtien, kuten hintatietojen, järjestysnumero html-koodissa saattoi vaihdella. Näin ollen eri testeissä tiettyjen elementtien hakemiseen ei voinut määritellä elementin XPath-polkua tietyn järjestysnumeron mukaan, koska sama elementti saattoi löytyä toisesta testistä eri paikasta käyttöliittymää. Tätä kyseistä ongelmatilannetta olin pohtinut viikkoanalyysissa 5.

Työstäessäni päivän aikana uusia laajakaistan tilausprosesseja pystyin työskennellessäni arvioimaan ja tutkimaan miten aikaisemmin tekemäni VPN-tilausprosessin testitapaukset olisi voinut tehdä paremmin, jotta ne olisivat voineet olla suoraan hyödynnettävissä myös muissa myyntijärjestelmän testeissä. Ohjelmoidessani laajakaistan tilauksen testitapauksia jouduin samalla myös muuttamaan osaa VPN:n testitapauksista, jotta kaikilla myyntijärjestelmän testitapauksilla olisi yhtenäinen rakenne ja logiikka.

Perjantai 21.7.2017

Tavoitteenani oli saada ohjelmoitua laajakaistan tilausprosessi kaapelitalouteen samaan vaiheeseen kuin muut työstämäni myyntijärjestelmän testit. Tämän jälkeen tarkoituksena oli aloittaa aikaisemmin työstämieni myyntijärjestelmän testien hiominen niin, että testin sisältämien testitapauksien rakenne olisi mahdollisimman yhtenäinen.

Koska laajakaistayhteyttä tilattaessa tuotteen mukana tilataan yhtäaikaisesti ylimääräinen palvelu, kuten Spotify ja tietoturva, jouduin parantelemaan hintatietojen tarkastusta laajakaistan tilauksissa. Tilattaessa laajakaistaa kaapelitalouteen lisäpalvelun lisäksi ostoskorista löytyi myös modeemilaitteen hintatiedot, jotka oli erikseen tarkistettava. Työstäessäni molempien laajakaistojen testejä yhtäaikaisesti en saavuttanut päivälle asettamaani tavoitetta. Näin ollen en ehtinyt aloittaa VPN-tilauksen testitapauksien uudelleen järjestelyäkään.

Vaikka olen Robot Framework automaatioita työstäessäni törmännyt useisiin erilaisiin ongelmiin etsiessäni web-elementtejä niiden XPath-polkujen avulla, tämän päivän aikana haastetta aiheuttivat eri tuotteiden ja alituotteiden hintatietojen hakeminen tuotteen tai alituotteen nimen mukaan. Aikaisemmin olin tottunut etsimään elementtejä liikkumalla html-koodin rakennetta alaspäin eli siirtymällä tietyn elementin alta löytyvään elementtiin //- ja /-komentojen avulla. Koska tällä kertaa tuotteen hinta- ja nimitiedot olivat samanarvoisia, vierekkäin sijaitsevia elementtejä, joilla oli sama vanhempi-elementti, jouduin liikkumaan XPath-polun avulla vastakkaiseen suuntaan html-koodissa kuin normaalisti. Yksinkertaisissa XPath-poluissa edetään html-koodissa portaita alaspäin, mutta tässä tapauksessa lähdin liikkeelle hintatiedon sisältävästä elementistä, siirryin sekä hinta- että nimitietojen sisältämien elementtien vanhempaan elementtiin ja siitä nimitiedot sisältävään elementtiin. Näin pystyin tässä tilanteessa tuotteen tai sen alituotteen nimen avulla löytämään sitä vastaavat hintatiedot web-käyttöliittymästä.

Olin aikaisemmin jo miettinyt onko XPathin avulla mahdollista edetä html-koodissa alhaalta ylöspäin myös mahdollista ja tämän päivän aikana oli hyvin hyödyllistä huomata, että oikeiden XPath-komentojen avulla se on. Samalla tavalla kuin siirryttäessä html-rakennetta ylöspäin, siirtyminen onnistuu myös niin sanotusti sivulta sivulle liikkumalla sisarelementistä toiseen. Oman Robot Frameworkin-koodin rakennetta kehittäessä ja parannellessa oppimista tapahtui myös löytämällä ja hyödyntämällä uusia tapoja kirjoittaa XPath-polkuja, joista ei ole muuta kuin hyötyä tulevia testejä ohjelmoidessani.

3.6.1 Viikkoanalyysi 6

Viikon aikana oppimiseni kehittyi kertauksen ja toistojen kautta työstäessäni uusia testitapauksia, joiden rakenteet olivat minulle jo aikaisempien testien ansiosta tulleet tutuiksi. Näin pystyin hyvin vertaamaan nykyistä ja aikaisempaa tapaani sekä kirjoittaa koodia että ratkaista siihen liittyviä ongelmatilanteita. Vertailun avulla pystyin kehittämään toimintatapojani ja ongelmanratkaisukykyäni eteenpäin pystyessäni selkeästi näkemään vanhasta koodistani, mitä voisin tällä kertaa tehdä paremmin. Näin pystyin myös huomaamaan ja välttämään mahdollisten virheiden kirjoittamista. Tämä toteutui hyvin työstäessäni tämän viikon loppupuolella uusia laajakaistan tilausprosesseja ja samalla verraten uutta koodiani viikon alussa tekemääni VPN-tilausprosessin koodiin.

Viikon aikana jouduin selvittämään useita eri Robot Frameworkin ohjelmoimiseen liittyviä ongelmia ja haasteita, joista osa, kuten sivustoon upotetun pdf-elementin tietojen hakeminen, jäivät valitettavasti vielä selvittämättä. Mielestäni tämän viikon suurin selvitettävä asia liittyi kuitenkin testaamani testiympäristön ja -sivustojen teknisiin ongelmiin.

Työstäessäni laajakaistan tilausta valokuitutalouteen, vastaani tuli ongelma, jossa testiympäristön ja -sivuston teknisten ongelmien takia saatavilla olevat laajakaistaliittymät eivät näkyneet sivustolla niin kuin niiden olisi kuulunut näkyä. Lähdin selvittämään tätä asiaa muilta myyntijärjestelmän testaamisesta vastaavilta henkilöiltä ja he epäilivät, että kyseessä on testiympäristöön ja -sivustoon liittyvät konfiguraatio-ongelmat, jonka vuoksi testisivusto ei saa haettua tarvittavia tuotetietoja muista järjestelmistä. He antoivat yhteystiedot henkilöille, joihin minun piti asiaan liittyen ottaa yhteyttä ja tehdä heidän johdollaan ongelmasta tiketti.

Vaikka ongelma olikin työskentelyäni hidastava ja sellainen, jota en pystynyt itse korjaamaan, loppujen lopuksi minun oli helppo lähteä selvittämänä asiaa. Tämä johtui siitä, että Helsingin kaupungin Kaupunginkansliassa tekemäni työharjoittelun ja Palvelukeskuksen puhelinvaihteessa tekemäni kesätöiden kautta opin hyviä käytännön tapoja ihmisten kanssa kommunikointiin. Näihin tapoihin kuului kommunikointi pikaviestinnällä, sähköpostitse, puhelimitse kuin kasvotustenkin joten kynnys aloittaa vastaan tulleen teknisen ongelman selvittäminen oli lähes olematon. Lähtiessäni selvittämään asiaa hyödynsin hyväksi katsomiani tapoja lähestyä ihmisiä jonkin asian tai ongelman tiimoilta. Minulle nämä toimintatavat ovat jo itsestäänselvyyksiä, mutta niitä ei tule pitää sellaisina. Tämä johtuu siitä, että kaikille ne eivät ole itsestäänselvyyksiä mistä johtuen on myös osattava soveltaa omia toimintatapojaan sen mukaan, kenen kanssa on milloinkin vuorovaikutuksessa.

Tällaisen, itsestään riippumattomat, tekniset ongelmat laittavat miettimään, miksi testausta suoritetaan vanhemmissa ympäristöissä, joita ei enää käytettä tai päivitetä aktiivisesti? Tilanne olisi eri jos edes vähänkin käytettäviä ympäristöjä päivitettäisiin yhtä aktiivisesti, koska näin erilaisten testiympäristöihin ja -järjestelmiin liittyvien ongelmien riskiä pystyttäisiin vähentämään kun testiympäristöjen ylläpidosta vastattaisiin paremmin. Näin testaajat välttyisivät myös teknisten ongelmien aiheuttamilta viivästymisiltä, koska kyseessä on testaajasta riippumaton ongelma. Tällaisissa tapauksisessa ongelmalle ei voi muuta kuin sen, että se pitää ilmoittaa eteenpäin.

Virheilmoituksia ja -raportteja kirjoittaessa on hyvä pitää mielessä se, että mitä huolellisemmin virheilmoitukset kirjoittaa ja mitä informatiivisempia ne ovat, sitä vähemmän ylimääräistä työtä koodaajat joutuvat tekemään. Kanerin, Bachin ja Pettichordin (2002, s.66) mukaan bugiraportteja voikin pitää myyntityökaluna, jolla vakuutetaan koodaajat käyttämään resurssejaan ja aikaansa löydetyn virheen etsimiseen ja korjaamiseen. Tästä syystä raporteista on siis selkeästi nähtävä millainen ongelma on kyseessä. Bugien raportointia

pääsin onneksi harjoittelemaan jo Kaupunginkansliassa manuaalista testausta suorittaessani, joten asia ei tullut

Virheilmoituksen siirtyessä henkilöltä toiselle ei voi kuitenkaan suoraan sanoa, kuinka kauan ongelman ratkeamisessa kestää, koska sitä käsittelevällä henkilöllä voi olla muitakin työtehtäviä suoritettavana. Tällöin työtehtävien suorittaminen tapahtuu niiden kriittisyyden ja niille annettujen aikamääreiden mukaisesti ja testaajan ongelma voi huonolla tuurilla joutua odottamaan jopa useita päiviä ongelman ratkeamiseksi eikä testaaja ei voi muuta kuin odottaa ja siirtyä työstämään seuraavia tehtäviä jos vain mahdollista. Onneksi minun tapauksessani huomaamani testiympäristöihin liittyvä ongelma korjaantui yön aikana eikä minun tarvinnut siirtyä hetkeksi tekemään jotain muita testejä, joka olisi sekoittanut taas ajatukset keskittymisen siirtyessä jatkuvasti asiasta toiseen.

Kuten jo viime viikollakin, tällaiset tilanteet laittavat myös miettimään sitä, miten itse ja muiden olisi tullut kommunikoida ennen varsinaisten testien aloittamista. Näin mahdolliset vastaan tulleet ongelmat ja kysymykset olisi alustavasti pystynyt käymään läpi ja informoimaan oikeat ratkaisut teknisiin ongelmiin ja siihen, kuka niistä esimerkiksi lomien aikana vastaa. On tietysti selvää, että kaikkia vastaantulevia ongelmia ei voi etukäteen arvioida ja odottamattomat ongelmat voivat olla jopa hyödyllisiä. Onneksi seuraavalla kerralla vastaavat virheet voi välttää helpommin.

3.7 Seurantaviikko 7

Maanantai 24.7.2017

Aloitettuani viime viikolla uusien myyntijärjestelmän testien ohjelmoinnin, tämän päivän tavoitteena oli jatkaa laajakaistatilauksen automatisointia kaapelitalouteen ja saada kyseinen testi samaan vaiheeseen kuin VPN:n ja toisen laajakaistatilauksen testit. Kyseessä oli tässä vaiheessa kuitenkin vain pelkän laajakaistayhteyden tilaaminen, koska päätin ennen varsinaisen testin työstämistä ohjelmoida laajakaistan ja tietoturvapalvelun tilaamisen erikseen ennen niiden yhdistämistä samaan testiin. Näin pystyisin hyödyntämään turvapalvelun tilaamisessa jo tekemääni VPN:n tilauksen koodia molempien ollessa VAS-palveluita ja joiden tilausprosessit ovat identtisiä.

Saavutin päivän tavoitteeni ja sain ohjelmoitua laajakaistayhteyden tilaamisen kaapelitalouteen tavoiteltuun vaiheeseen. Tämän jälkeen päädyin korjaamaan VPN-tilausprosessia lisäämällä testiin omia testitapauksiaan erilaisten tarkistuksien suorittamiseksi, aivan kuten verkkokaupan testien kohdalla. Korjattuani VPN-tilauksen koodia, siirryin työstämään

kaapelitilauksen varsinaista testiä, jossa laajakaistayhteyden ohella tilataan myös tietoturvapalvelu.

Tiistai 25.7.2017

Päivän tavoitteena oli jatkaa kaapeliyhteyden ja tietoturvapalvelun yhteistä tilausprosessia ohjelmointia. Työskentely ei kuitenkaan sujunut täysin ongelmitta, koska kuluneen päivän suurimmaksi ongelmaksi nousi desimaaliluvun muuttaminen kahden desimaalin tarkkuuteen niin, että myös desimaapilkun jälkeiset ylimääräiset nollat olisivat näkyvissä.

Tämä kysymys oli mietityttänyt minua jo aikaisempien viikkojen aikana, mutta nyt vasta siihen liittyvät ongelmat tulivat varsinaisesti vastaan. Ongelmana oli se, että muutettaessa tekstijonoa numeroiksi tai annettaessa muuttujalle numeroarvo, Google Groups-sivuston kirjoituksen mukaan Python-ohjelmointikieli, joka toimii Robotin logiikan takana, pudottaa automaattisesti mukaan ilmoitetusta desimaaliluvusta desimaalipilkun jälkeiset ylimääräiset nollat pois (2016). Tästä syystä esimerkiksi luvusta 32,90 tulee automaattisesti 32,9. Tämä aiheutti ongelmia verratessa tilattavan tuotteen hintatietoja keskenään, koska hintatietojen vertailu tapahtui vertaamalla kahta tekstijonoa, en numeroarvoa, keskenään. Olin aikaisemmilla viikoilla päätynyt tähän ratkaisuun, koska hintatiedot löytyivät testattavilta sivustoilta tekstijonoina eikä näin ollen hintatietoja tarvinnut muuttaa aina erikseen numeroarvoiksi.

Aluksi ongelma vaikutti helpolta ratkaista, mutta ratkaisu ei ollutkaan niin yksinkertainen kuin oletin. Ensimmäiseksi kokeilin muuttaa tekstijonot numeroiksi Robot Frameworkin omien avainsanojen sekä Javascriptin-komentojen avulla, mutta tuloksetta. Tutkittuani asiaa lisää internetistä, huomasin, että moni Python-ohjelmointikielen käyttäjä, oli myös törmännyt samanlaiseen ongelmaan. Stack Overflow-sivustolta tähän ongelmaan löytyi ratkaisu, jonka mukaan Pythonilla oli ajettava koodirivi, joka muotoili halutun numeroarvon haluttuun desimaalimuotoon (2017).

Ratkaisun löydyttyä minun oli enää vain selvitettävä, miten Python koodia pystyi ajamaan suoraan Robotin Frameworkissa samalla periaatteella kuin Javascriptia-koodia. Tutkittuani Robot Frameworkin BuiltIn-nimistä kirjastoa tarkemmin huomasin, että jo aikaisemmin käyttämälläni Evaluate-avainsanalla pystyi myös ajamaan suoraan Python-koodia (Robot Framework Foundation, 2017a). Tämän avainsanan avulla sain ajettua tarvittavan Python koodin halutun numeroarvon muotoilemiseksi niin, että myös desimaalipilkun jälkeiset, ylimääräiset nollat olivat näkyvissä. Ongelman ratkettua sain työstettyä testiä eteenpäin ja saavutettua päivälle asettamani tavoitteen.

Keskiviikko 26.7.2017

Edettyäni eilen kaapeliyhteyden ja tietoturvapaketin tilauksen automatisoinnissa, asetin tämän päivän tavoitteeksi saada kyseisen testi samaan vaiheeseen kuin muut työstämäni myyntijärjestelmän testit. Kun testit olisivat samassa vaiheessa, pystyisin seuraavaksi aloittamaan sivuun upotetun pdf-elementin tarkastelemisen.

Työstämällä sekä uusia että vanhoja testitapauksia ja ottamalla mallia jo aikaisemmin tekemistäni testeistä, sain ohjelmoitua kaapeliyhteyden ja tietoturvapalvelun tilauksen haluttuun vaiheeseen ja saavutin päivälle asettamani tavoitteen. Sain myös viimein vastauksen TV-palvelun tilaukseen liittyviin, kuudennella raportointiviikolla vastaan tulleisiin kysymyksiin, erään kollegani palatessa lomalta. Ongelmana oli, että käyttämälläni testisivustolla ei näkynyt oikeaa TV-palvelun tuotevaihtoehtoa. Sain kollegaltani varmistuksen, että tässä testissä riittää että TV-palvelusta tilataan mobiiliversio, joka näkyi sivulla virheestä huolimatta. Näin ollen pääsin vihdoin työstämään viimeistä myyntijärjestelmän testiä eteenpäin.

Olin jo kuudennella raportointiviikolla aloittanut TV-palvelun tilaavan testin työstämistä, joten testin ohjelmointia oli hyvin helppo jatkaa. Käytännön takia tein TV-palvelun tilauksen tietyt vaiheet omina testitapauksinaan ja niitä valmiita testitapauksia, joita pystyin hyödyntämään TV-palvelun tilaamisessa muokkasin vain niin, että ne toimivat myös tässä testissä. Seuraamalla muiden myyntijärjestelmän testien ja testitapauksien rakennetta, sain nopeasti iltapäivän aikana automatisoitua TV-palvelun tilauksen samaan vaiheeseen, kuin missä muutkin myyntijärjestelmän tilausprosessit olivat.

Aloittaessani tutkimaan, miten Robotilla pystyisi lukemaan verkkosivuun upotetun pdf-tiedoston tietoja, sainkin kuulla, että minun ei nyt tarvitsisikaan keskittyä pdf-tiedoston sisällön tarkasteluun. Tässä vaiheessa riitti pelkän pdf-tiedoston olemassa olon tarkistus. Tähän ratkaisuun päädyttiin sen takia, koska pdf-tiedostomuotoa olevan sopimuksen tiedot haetaan jo tilausprosessin aikaisemmassa vaiheessa, joten riittäisi, että sopimusta edeltävän vaiheen tiedot tarkistettaisiin. Koska olin jo ohjelmoinut kyseisten tietojen tarkistuksen ja jäljelle jää enää vain tilausvahvistuksen tietojen tarkastaminen sekä koodin hienosäätäminen.

Torstai 27.7.2017

Saatuani tiistaina selvitettyä, miten Robot Frameworkissa pystyy muuttamaan numeroarvon kahden desimaalin sisältävään muotoon, asetin tämän päivän ensimmäiseksi tavoitteeksi korjata muutama yhden verkkokaupan testin sisältämistä testitapauksista. Kyseessä oli testi mobiililaitteen tilaamiseksi kertamaksulla. Kyseisistä testitapauksista puuttui vaihe, jossa tilattavan tuotteen hinta ja postikulut lasketaan yhteen. Aikaisemmin hain kokonaishinnan Robotin avulla suoraan verkkokaupan sivulta, laskematta sitä itse. Tämän takia päätin korjata kyseisen testitapauksen löydettyäni oikean tavan numeroarvojen muotoiluun ja yhteenlaskuun. Saman aikaisesti myös siistin verkkokaupan testien koodia muun muassa muuttamalla testitapausten nimiä entistä selkeämmiksi ja kuvaavimmiksi ja lisäämällä koodiin tarvittavia kommentteja.

Saatunani verkkokaupan testien koodia hiottua, siirryin hiomaan myyntijärjestelmän laajakaistan tilauksien koodia lisäämällä sinne testitapauksen, joka tarkistaa, että onko asiakkaan ilmoittamassa osoitteessa jo käytössä laajakaistapalvelu. Jos kyseisessä osoitteessa on jo kyseinen palvelu käytössä, Robotti ajaa avainsanan Fatal Error, mikä pysäyttää testin virheilmoitukseen (Robot Framework Foundation, 2017a). Samalla Robotti ilmoittaa testaajalle, että valitussa osoitteessa on jo käytössä laajakaistapalvelu, joten testaaja tietää vaihtaa tilauksen osoite.

Päivän aikana oppiminen tapahtui kertauksen kautta hyödyntäessäni tämän ja viime viikon aikana myyntijärjestelmän automatisoinnissa oppimiani asioita korjatessani ohjelmoimiani koodeja. Uusia asioita opin kun kävimme opastetusti läpi, miten esimerkiksi myyntijärjestelmän kautta tehty tilaus löydetään ja poistetaan CRM-järjestelmästä. Tämä oli hyvin hyödyllistä, koska nyt pystyin ajamaan myyntijärjestelmän testitilauksia samaan osoitteeseen uudestaan poistamalla aikaisemman tilauksen CRM:n kautta, koska esimerkiksi laajakaistan voi tilata tietylle käyttäjälle tiettyyn osoitteeseen vain kerran.

Ongelmaksi kuitenkin nousi se, että vaikka tilaukset oli CRM:n kautta peruutettu, myyntijärjestelmän web-käyttöliittymän mukaan osa tilauksista oli poistosta huolimatta edelleen käsittelyssä. Kyse oli kollegoideni mukaan siitä, että tilauksen tiedot eivät olleet vielä ehtineet synkronoitua eri järjestelmien välillä. Tarkistan tilanteen vielä huomenna uudestaan, koska kollegoideni mukaan tietojen synkronoinnissa ei pitäisi mennä kuin muutamasta minuutista muutamaan tuntiin.

Perjantai 28.7.2017

Viikon viimeisen raportointipäivän tavoitteena oli tarkistaa CRM-järjestelmästä, että myyntijärjestelmän kautta tehdyt automatisoidut tilaukset olivat menneet läpi. Tässä tarkistuk-

sessa hyödynsin eilen oppiminani taitoja liittyen tilauksen peruuttamiseen CRM-järjestelmän kautta. Saatuani onnistuneesti varmistettua, että kaikki myyntijärjestelmän tilausprosessit olivat menneet läpi, jäljelle jäi ohjelmoida Robottiin uusi testitapaus. Testitapaus tarkistaa, että tilauksen lähettämisen jälkeen myyntijärjestelmän käyttöliittymässä lukee tilatun tuotteen kohdalla teksti, joka ilmoittaa tilauksen olevan käsittelyssä. Sain XPath-poluilla haettua jokaisen tuotteen kohdalla ne elementit, joissa tilauksen käsittelystä ilmoittava teksti löytyy, mutta varsinaisia testitapauksia en kuitenkaan vielä päivän aikana saanut tehtyä. Työn helpottamiseksi käytössäni olevista versionhallinnan Robot Framework-esimerkeistä löytyy valmis rakenne kyseisten testitapausten ohjelmoimiseksi.

Eilen vastaan tullut ongelma liittyen tilauksen tietojen synkronointiin eri testijärjestelmien välillä ratkesi viimein, mutta ei täysin haluamallani tavalla. Näyttäisi siltä, että kun tilaus on peruttu, tietojen synkronoinnissa voi mennä jopa yli 24 tuntia. Tämä siis tarkoittaa sitä, että ennen kuin tietyt myyntijärjestelmän tilausautomaation voi ajaa uudestaan samoilla tiedoilla kuin aikaisemmassakin testissä, joudun odottamaan jopa 24 tuntia.

3.7.1 Viikkoanalyysi 7

Seitsemännen raportointiviikon aikana oppimiseni Robot Frameworkin sekä testattavien järjestelmien parissa kehittyi taas askeleen eteenpäin. Muutamien viime viikkojen teemana ollut oman koodin laadun ja rakenteen tarkastelu sekä parantelu olivat myös tällä viikolla läsnä vaikka tämän viikon oppiminen keskittyi uusien asioiden oppimiseen. Tämä johtui siitä, että testiautomaatioita ohjelmoidessa vastaan tuli ongelmia, joita en pystynyt nykyisillä tiedoillani ratkaisemaan. Nämä vastaan tulleet ongelmat, kuten numeroarvon muuttaminen kahden desimaalin muotoon, olivat oppisen kannalta hyvin hyödyllisiä, koska ne pakottivat etsimään uusia ratkaisua, jotta testin työstämisen jatkaminen olisi mahdollista.

Tämän viikon loppupuolella ongelmia tuotti tehtyjen tilauksien tietojen synkronointi testiympäristön eri järjestelmien välillä. Kun esimerkiksi myyntijärjestelmän kautta tehty tilaus oli poistettu CRM-järjestelmästä, muutaman tunnin sijasta meni kokonainen vuorokausi ennen kuin tieto tilauksen peruuttamisesta kulkeutui myyntijärjestelmään asti. Tämä taas johti siihen, että tiettyjä testejä ei voinut ajaa samoilla tilaustiedoilla uudestaan, koska esimerkiksi laajakaistaa ei ole mahdollista tilata samalla testikäyttäjällä samaan osoitteeseen uudestaan.

Koska tietojen synkronointiin kului paljon aikaa, se hidasti ja hankaloitti testiautomaatioiden ajamista. Oletuksena oli, että saman testin voisi ajaa samoilla tiedoilla, kuten samalla

tilausosoitteella ja tuotteella, nopeasti uudestaan tilauksen poistamisen jälkeen. Näin saman päivän aikana olisi ollut mahdollista toistaa sama testi ongelmitta useampaan kertaan. Mutta tilauksen käsittelyn, eli tässä tapauksessa tilauksen tietojen synkronoinnin, ollessa vielä kesken, tiettyjä testejä ei voinut ajaa samoilla tiedoilla uudestaan.

Viiveet tilauksen tietojen synkronoinnissa toi mukanaan rajoitteita, jotka mielestäni laskivat tilausprosessin testattavuutta ja toistettavuutta huomattavasti. Tämä johtui siitä, että ennen jokaista tilausta Robotin muuttujiin oli aina määritettävä, mitkä olivat seuraavassa testiajossa käytettävät tilaustiedot. Tämän takia myös tehtyjen tilauksien, ja ylipäätänsä ajettavien testien, hallinta oli mielestäni myös turhan epäkäytännöllistä. Jokaisessa testiajossa esimerkiksi muuttuvan tilausosoitteen myöstä, tilauksia lähti aina uuteen osoitteeseen luoden useita tilausrivejä. Jos testit olisi voinut ajaa uudestaan samoilla tiedoilla mahdollisimman nopeasti uudestaan, tilauksen tietoja ei olisi aina tarvinnut erikseen manuaalisesti muuttaa mikä olisi nopeuttanut testin suorittamista ja nostanut järjestelmän testattavuutta. Tällöin jatkuvasti muuttuvista testiosoitteista ei olisi myöskään tarvinnut pitää niin tarkkaa kirjaa.

Raportointiviikkojen 4-7 aikana kollegoideni kesälomat pakottivat minut kohtaamaan vastaan tulleet ongelmat täysin itsenäisesti. Sekä asioiden selvittäminen, että erilaisten ongelmien ratkaiseminen oli tuolloin täysin omalla vastuullani ja parhaimmillaan joidenkin ongelmien selvittämisessä saattoi kulua jopa muutama päivä ennen kuin sain ne ratkaistua. Tämä oli oppisen kannalta kuitenkin erittäin antoisaa, koska ongelmien ratkaisemiseksi minun täytyi jatkuvasti haastaa itseäni etsimää ja kokeilemaan uusia tapoja näiden ongelmien ratkaisemiseksi.

Tärkeintä ongelmia ratkaistaessa oli löytää toimiva ratkaisu ongelmaan, vaikka se ei olisi ollutkaan ehkä kaikista paras mahdollinen ratkaisu. Näin en vastaan tulleen ongelman takia jäänyt jumiin eikä ongelma hidastanut töiden etenemistä entuudestaan, vaikka en olisi löytänytkään tilanteeseen ”parhaiten” sopivaa ratkaisumallia. Jo tehtyyn koodiin oli helppo palata uudestaan jos edetessä vastaan tuli toimivampia ratkaisuja tai toimintamalleja kuten Robotin tapauksessa uusia avainsanoja ja kirjastoja. Näin ollen koodia oli helppo korjata myös myöhemmin oman tietoperustan ja osaamisen kertyessä. Tämä liittyy vahvasti koodin tarkasteluun kriittisemmästä näkökulmasta sen laadun ja toimivuuden parantamiseksi.

Vaikka ongelmien ratkaiseminen täysin itsenäisesti oli oppimisen kannalta erittäin hyvä, vastaan tuli myös omasta toiminnasta riippumattomia ongelmia, kuten aikaisemmin mainitsemiani testisivustoihin ja -ympäristöön liittyviä teknisiä ongelmia tai tilauksen tietojen

synkronoinnin hitaus eri testijärjestelmien välillä. Näihin ongelmiin ei itse oikein muuten voinut vaikuttaa kuin yrittää informoida kyseisistä ongelmista eteenpäin. Tämä oli kuitenkin melko hidasta henkilökunnan lomien takia ja jouduin välillä suoraan siirtymään testistä toiseen kun aikaisemmin työstämääni testiä ei voinut jatkaa.

3.8 Seurantaviikko 8

Maanantai 31.7.2017

Saatuani ne verkkokaupan ja myyntijärjestelmän testit, joita pystyin työstämään testiympäristöihin liittyvistä ongelmista huolimatta, ohjelmoitua, tämän päivän tavoitteena oli valmiiden koodien hiominen. Valmiiden koodien hiomiseen kuului muun muassa yhtenäisten lokimerkintöjen ulkoasun kehittäminen. Kyseessä on siis Robotin PyCharmiin kirjoittama konsoliloki ja tarkoituksena olisi, että kyseiset lokimerkinnät olisivat kaikissa testeissä samalla tavalla muotoiltuja. Koodin hiomiseen kuuluivat myös yleisesti koodin ulkoasun ja rakenteen yhtenäistäminen muun muassa niin, että avainsanat ja testitapaukset olisivat kaikki kirjoitettu isolla alkukirjaimella.

Saavutettuani päivän tavoitteen jatkoin viime perjantaina kesken jääneiden myyntijärjestelmän testien sisältämien testitapauksien työstämistä. Näiden testitapausten tarkoituksena oli tarkistaa, että tuotteen tilaus on käsittelyssä tilauksen lähettämisen jälkeen. Kyseisiä testitapauksia työstäessäni oppiminen tapahtui malliesimerkkiä hyödyntämällä, mikä on ollut yksi pääasiallisista oppimismenetelmistäni tutustuessani Robot Frameworkiin aikaisempienkin viikkojen aikana.

Päivän päätteeksi aloitin kolmannen ja viimeisen testattavan järjestelmän testiautomaatioiden kirjoittamisen. Kyseessä on jo Telian asiakkaana olevan käyttäjän omat sivut, jonka kautta asiakas pystyy muun muassa hallitsemaan omia liittymiään ja muita käytössään olevia Telian palveluita. Tässä järjestelmässä tehtäviin testeihin kuului seuraavien prosessien automatisointi: käyttäjän laajakaistan nopeuden nostaminen ja laskeminen, mobiililiittymän laskutustietojen muuttaminen sekä uusien kanavapakettien lisääminen voimassa olevaan TV-palvelutilaukseen.

Päätin aloittaa työskentelyn mobiililiittymän laskutustietojen muuttamisesta ja käytin siinä mallina eritoten tekemiäni verkkokaupan testejä, asiakkaan sivujen käyttöliittymän ollessa hyvin samanlainen verkkokaupan käyttöliittymän kanssa. Myös tässäkin automaatiossa käytin hyväkseni automaatiotiimin versionhallinasta lataamiani malleja ja esimerkkejä, joista löytyi valmiiksi tehtynä asiakkaan omien sivujen testien ensimmäiset vaiheet, kuten

sivuston avaaminen. Iltapäivän aikana ehdin ensimmäisen omien sivujen testin ohjelmoinnissa vasta alkuvaiheeseen päivän ollessa työn täytteinen.

Tiistai 1.8.2017

Tämän päivän tavoitteena oli käydä läpi tiimin testiautomaatiovastaavan kanssa raportointiviikkojen 4-7 aikaansaannokseni lyhyessä koodikatselmuksessa. Katselmuksen tarkoituksena oli käydä läpi koodini yleisilmettä, testien ja testitapauksien rakennetta sekä yleisesti katsoa testattavien järjestelmien testien tilanne. Kokouksen lopuksi tarkoituksena oli asettaa tavoitteet raportointiviikolla 9 olevaa katselmusta varten, johon myös muiden tiimin jäsenten on tarkoitus osallistua.

Saimme mainiosti kollegani kanssa noin kahden tunnin aikana käytyä ohjelmoimaani koodia läpi esitellessäni ja selittäessäni miten kukin testi ja testitapaus on rakennettu, miten ne toimivat ja mitä avainsanoja olen niiden ohjelmoimisessa käyttänyt. Demosin myös muutamia tekemiäni testejä, kuten mobiililiittymän ja puhelimen tilaamisen verkkokaupasta. Pitämämme kokous oli hyvin hyödyllinen, koska sain sen aikana paljon rakentavaa palautetta aikaansaannoksistani. Palautteen kautta sain selville mikä oli tehty hyvin, missä oli vielä hiottavaa ja minkä voisi yrittää tehdä vielä eri tavalla. Tämän päivän aikana oppiminen tapahtuikin pääsääntöisesti saamani palautteen kautta ja nähdessäni itsekin koodini kommenttien kautta täysin uudesta näkökulmasta.

Katselmuksen jälkeen sain tehtäväkseni tehdä koodiini muutamia korjauksia kuten muutamien kovakoodattujen arvojen muuttamisen omiksi muuttujikseen. Näin tilattavan tuotteen tietoja on mahdollista muuttaa testitapausten yläpuolelta testitasolta ja samalla testien toistettavuus paranee. Poistamalla kovakoodatut arvot koodista, tilattavaa tuotetta pystyy vaihtamaan helposti ja samalla muuttujien arvot ovat kaikkien testitapausten ja avainsanojen käytettävissä. Tavoitteena on saada nämä korjaukset tehtyä huomisen aikana ja sen jälkeen jatkaa omien sivujen testien työstämistä.

Keskiviikko 2.8.2017

Päivän tavoitteena oli tehdä ohjelmoimaani koodiin eilen pidetyn koodikatselmuksen myötä esiin tulleet korjaustarpeet. Korjaukset koskivat valmiita verkkokaupan ja myyntijärjestelmän testejä, joista löytyi pientä hiottavaa kuten koodin sisälle kovakoodattujen arvojen poistaminen ja käyttämieni XPath-polkujen tarkistaminen. Jos aikaa jää jäljelle, näiden korjaustöiden jälkeen tarkoituksenani on siirtyä jatkamaan asiakkaan omien sivujen testien ohjelmoimista.

Saavutin asettamani tavoitteet vain osittain, koska en saanut kaikkia tarvittavia korjauksia päivän aikana tehtyä. Pääasiallisesti tein tarvittavia korjauksia poistamalla koodista kovakoodattuja arvoja kuten joitakin hintatietoja ja tilattavien tuotteiden nimiä. Näin ollen esimerkiksi tilattavaa tuotetta ei voinut muuttaa testitasolta. Näille kovakoodatuille arvoille oli luotava omat muuttujansa, jotta kyseisiä arvoja pystyisi tarvittaessa muuttamaan testitasolta tilausprosessien toimivuuden testaamiseksi myös eri tuotteilla.

Muuta korjattavaa aikaansaamissani koodeissa oli muutamien elementtien XPath-polkujen korjaaminen. Koodiin oli jäänyt vielä muutamia elementin järjestysluvun mukaan määriteltyjä XPath-polkuja, jotka huomasin varsinkin kun verkkokaupan testisivustoa päivitettiin. Aikaisemmin järjestysluvun mukaan määritellyt elementit eivät enää sijainneet samassa paikassa sivua kuin aikaisemmin ja näin ollen Robotin ei niitä pystynyt paikantamaan. Tämän takia muokkasin XPath-polkuja niin, että Robotti pyrkii paikantamaan tarvittavat elementit pääsääntöiesti niiden nimen tai id-tunnisteen avulla. Samankaltainen ongelma oli tullut vastaani myös raportointiviikolla kuusi.

Päivän aikana oppiminen tapahtui tekemiäni virheitä tutkimalla ja niitä korjaamalla, mikä oli oppisen kannalta hyvin hyödyllistä. Näin pääsin sekä arvioimaan että huomaamaan, minkälaista kehitystä Robot Frameworkin ohjelmointitaidoissani oli tapahtunut viimeisen seitsemän raportointiviikon eli kahdeksan ja puolen työviikon aikana. Loppuviikon tavoitteena on yrittää jatkaa valmiiden koodien hiomista niin, että ne ovat valmiit ja korjatut ensi viikon koodikatselmusta varten.

Torstai 3.8.2017

Päivän tavoitteena oli jatkaa verkkokaupan ja myyntijärjestelmän testien korjauksien tekemistä. Pääosin aika meni verkkokaupan koodia korjaillessa muutostöiden osoittautuessa työläämmiksi kuin aluksi kuvittelin. Alkuperäisenä tavoitteena oli saada molempien järjestelmien testaamiseen liittyvät korjaukset valmiiksi jo keskiviikon aikana. Nyt tavoitteena oli saada ainakin verkkokaupan testit korjauksineen valmiiksi. Saavutin päivälle asettamani tavoitteet ja sain valmiiksi verkkokaupan testiautomaatiot korjauksineen lukuun ottamatta uuden mobiililaitteen tilaamista osamaksulla, jota en pystynyt vieläkään ohjelmoimaan loppuun. Tähän vaikutti edelleen sama tekninen virhe, joka ilmeni jo muutama raportointiviikko sitten aloittaessani kyseisen testin ohjelmoimista.

Korjasin aikaansaamaani koodia esimerkiksi lisäämällä siihen uusia tarvittavia testitapauksia. Esimerkiksi jos tilattavalla tuotteella on voimassa oleva alennus tai tarjous, Ro-

botti tarkistaa ostoskoriin siirryttäessä, että nykyinen eli alennettu hinta ei ole suurempi kuin tilattavan tuotteen normaalihinta. Jos sivusto näyttää normaalin hinnan suuremmaksi kuin nykyisen hinnan, Robotti pysäyttää ajettavan testin ja antaa kyseiseen tilanteeseen määritellyn virheviestin. Alun perin olin ohjelmoinut mobiililiittymän tilaavan testin niin, että Robotti hakee ja tallentaa sekä tuotteen nykyisen, että normaalin hinnan. Kävimme asiaa kollegani kanssa vielä tarkemmin läpi ja tässä vaiheessa testien ohjelmointia riitti, että Robotti vain vertaa normaalia hintaa tallennettuun voimassaolevaan hintaan. Robotin ei siis tarvitsisi enää tallentaa myös tuotteen normaalia hintaa kuten aikaisemmin.

Tein muutoksia myös testeihin, joissa tilataan mobiililaite ja laajakaista määriteltyyn osoitteeseen. Nyt testaaja pystyy testitasolle määriteltyjen muuttujien arvoja muokkaamalla valita, minkä matkapuhelimen ja minkä laajakaistatuotteen ja siihen kuuluvan modeemilaitteineen hän tilaa. Näin testit voi suorittaa useilla eri arvoilla, mikä parantaa järjestelmän testattavuutta. Nyt on mahdollista tarkistaa toimivatko kyseiset testit oletetusti myös muiden tilattavien tuotteiden kohdalla. Samalla periaatteella korjasin myös mobiililaitteen tilauksen osamaksulla niin, käyttäjä voi erikseen määrittää, valitaanko tilattavalle laitteelle 12, 24 vai 36 kuukauden maksuaika.

Oppiminen tapahtui päivän aikana aikaansaamiani koodeja ja havaitsemiani ”virheitä” tutkimalla sekä niitä korjaamalla. Oppimisen kannalta oli hyvin palkitsevaa huomata, miten oman koodin ohjelmoiminen on parantunut kuluneiden viikkojen aikana uusia toimintatapoja ja Robot Frameworkin avainsanoja sisäistämällä. Huomatessani konkreettisesti itse ohjelmoimistani koodeista kehitykseni, pystyin samalla myös hyödyntämään oppimiani asioita korjatessani ja parannellessani tekemääni koodia.

Perjantai 4.8.2017

Saatuani eilen verkkokaupan testien koodit viimeisteltyä ja korjattua, tämän päivän tavoitteena oli viikon teeman mukaisesti tehdä myös myyntijärjestelmän testeihin tarvittavat lisäkorjaukset. Korjauksien jälkeen tarkoituksenani oli jatkaa maanantaina aloittamani, asiakkaan omien sivujen kautta tehtävän mobiililiittymän laskutustietojen korjaamisen automatisointi.

Saavutin päivälle asettamani tavoitteen saamalla myyntijärjestelmän testien koodit korjauksineen valmiiksi. Näihin korjauksiin kuului muun muassa mahdollisuus valita tilattava modeemilaite laajakaistaa tilattaessa samalla tavalla kuin verkkokaupan vastaavassa testissä. Muina korjauksina lisäsin esimerkiksi tarkistuksen, joka tarkistaa, että lisättäessä

uusi tuote ostoskoriin, ostoskori-ikonista löytyvä numero vastaa ostoskorissa olevien tuotteiden lukumäärää.

Haastetta korjauksien tekemisessä toi varsinkin uuden hintatarkistuksen tekeminen, koska osassa myyntijärjestelmän tuotesivuista tuotteen hinta vastasi tuotteen omaa, yksilöllistä hintaa ja taas osassa sivuista hinta vastasi tuotteen ja sen alituotteen, kuten Spotify-palvelun, yhteishintaa. Tämän takia minun oli ohjelmoitava ehto, joka ensin tarkistaa minkä tuotteen tilausprosessi on kyseessä ja sen mukaan käyttää hintavertailussa joko tuotteen yksilöllistä hintaa tai tuotteen ja sen alituotteen yhteishintaa.

Oppiminen tapahtui tämän päivän aikana kertaamalla verkkokaupan testeihin tekemiäni korjauksia ja hyödyntämällä niitä ohjelmoidessani uusia korjauksia myyntijärjestelmän testeihin. Kertauksen kautta pystyin samalla itse myös katselmoimaan omaa koodiani sillä silmällä, että olisivatko jotkut tekemäni korjaukset mahdollisesti voinut tehdä entistä paremmin tai erilaisia toimintatapoja hyödyntäen.

3.8.1 Viikkoanalyysi 8

Kahdeksannen raportointiviikon pääpainotteisena teemana oli verkkokaupan ja myyntijärjestelmän testien koodien korjaaminen ja hiominen tiistai iltapäivällä järjestetyn koodikatselmuksen pohjalta. Kollegani kanssa käymässäni katselmuksessa pureuduimme raportointiviikkojen 4-7 aikana tekemiini Robot Framework testeihin niin testi kuin myös testitapaustasolla. Tämä tarkoitti sitä, että ensin kävimme läpi minkälaisista ja nimisistä testitapauksista kukin testi koostui ja tämän jälkeen siirryimme niin sanotusti tasoa alemmas tutkimaan muun muassa minkälaisista avainsanoista ja muuttujista kukin testitapaus rakentui.

Koodikatselmuksen aikana ajoimme läpi myös muutaman testin, jotta kollegani näki myös miten nämä testit toimivat käytännössä. Samalla keskustelimme myös mahdollisista koodiin tehtävistä korjauksia ja silmäilimme sekä uusia avainsanoja että kirjastoja, joita voisi kokeilla ja hyödyntää eri tilanteissa. Tällaisia tilanteita ovat esimerkiksi ne, joissa annettu avainsana ei jostain syystä toimikkaan ja Robotti antaa virheilmoituksen pysäyttäen samalla käynnissä olleen testin virheeseen vaikka annetun avainsanan pitäisikin varmuudella toimia.

Viikon aikana suurin oppiminen tapahtui koodikatselmuksen aikana vaikka koko viikon aikana tapahtui jatkuvasti oppimista työstäessäni vanhaa ja uutta koodiani. Samalla pystyin oivaltamaan miten pystyisin sekä parantamaan koodiani että myös hyödyntämään

aikaisempia tuotoksiani tuoreimmissa koodeissani ja toisinpäin. Toisen henkilön tarkastellessa ja tutkiessa omaa koodiani pystyin itsekin huomaamaan virheitä ja muita huomion arvoisia ratkaisuja, joita en ollut itse aikaisemmin huomannut. Näin pystyimme keskustelemaan niistä yhdessä ja tarkastelemaan, miten ne tulisi korjata laadullisesti ja toiminnallisesti parhaan mahdollisen Robot Framework-koodin saavuttamiseksi. Saamani rakentavan palautteiden ja huomioiden kautta opin, miten Robotin koodia pystyy ohjelmoimaan entistä paremmin ja mitä asioita olisi hyvä huomioida omaa koodia työstäessä.

Kun omaa koodia työstää tarpeeksi kauan, voi siihen turtua. Tällöin ohjelmoija ei itse enää välttämättä kykene huomaamaan tekemiään virheitä, koska hänen silmänsä ovat ”sokeutuneet” omasta koodista. Siksi olisi hyödyllistä, että työn edetessä omaa koodia katselmoisivat välillä tuoreet silmäparit (Kaner, Bach & Pettichord, 2002, s.28). Hyvänä esimerkkinä tästä oli tämän viikon loppupuolella tapahtunut ongelmatilanne verkkokaupan hintatietojen vertailussa, jossa Robotti ilmoitti virheellisesti, että tuotteen nykyinen hinta olisi suurempi kuin sen normaali hinta. Itse en huomannut omassa koodissani virheittä, mutta kollegani huomasi, että osa hintatietoja sisältävistä muuttujista oli heittomerkkien sisällä. Nämä ylimääräiset merkit olivat muuttaneet muuttujien hintatiedot numeroarvoista tekstijonoiksi, joka johti siihen, etteivät ne olleet keskenään verrattavissa.

Robot Framework-koodeistani saamani palaute mielessäni pystyin entistä kriittisemmin ja eri näkökulmista tutkimaan omaa koodiani tehdessäni siihen korjauksia ja hioessani sitä. Tämä oli oppimisen kannalta hyvin palkitsevaa, koska pystyin selkeästi huomaamaan oman kehitykseni aikaansaamistani koodeista sekä samalla myös punnitsemaan miten pystyisin entuudestaan parantamaan Robot Frameworkin ohjelmointitaitojani. Näin ollen pystyn entistä määrätietoisemmin asettamaan itselleni tavoitteita oppimisen suhteen ja myös hieman etukäteen tarkastelemaan, minkä osa-alueiden työstämistä minun tulisi harjoitella entistä enemmän.

Kahdeksannen raportointiviikon aikana vastaan tulleet ongelmat liittyivät valmiiden testitapauksien korjaamiseen. Korjauksien tekemisessä ongelmia tuotti se, että valmiita testitapauksia muokattaessa sain olla hyvin tarkkana siitä, että en lisäyksilläni estänyt muiden testitapausten ja näin ollen koko testien toimintaa eli ”rikkonut” muuta koodia. Korjaustöiden vaatima pikkutarkkuus ja huolellisuus tekivät työn etenemistä hidasta, koska tehtyjen korjauksien jälkeen oli aina tarkistettava, että kaikki korjattua testitapausta käyttävät testit toimivat edelleen odotetusti. Tällaisia testitapauksia olivat järjestelmien globaalit testitapaukset, joita pystyi hyödyntämään testistä riippumatta. Eteneminen oli puolestaan taas nopeampaa, jos oli vain varmistettava, että ainoastaan yhdessä testissä käytettävä testitapaus eli testikohtainen testitapaus toimisi normaalisti.

3.9 Seurantaviikko 9

Maanantai 7.8.2017

Saatuani viime viikon loppupuolella myyntijärjestelmän testit valmiiksi, tämän päivän tavoitteena oli tarkistaa tekemäni koodi vielä kertaalleen läpi ja ajaa kaikki myyntijärjestelmän testit läpi niiden toimivuuden testaamiseksi. Jos iltapäivästä jää vielä aikaa, tarkoituksenani on jatkaa viime viikon alussa aloittamani asiakkaan omien sivujen testejä.

Ennen testien läpiajoja tarkistin oliko koodissa vielä tarvetta pienille korjauksille. Läpiajojen jälkeen hioin koodia vielä hieman lisää saadakseni sen ulkoasun vastaamaan tiimimme yhteisestä versionhallinnasta löytyvien koodien ulkoasua. Tästä johtuen muutin kaikki testitapauksien ulkopuolella, esimerkiksi testitasolla tai muissa tiedostoissa, määriteltävät muuttujat isoilla kirjaimilla kirjoitetuksi. Ne muuttujat taas, jotka määritellään testitapausten sisällä, muutin pienillä kirjaimilla kirjoitetuiksi.

Sain päivälle asettamani tavoitteen saavutettua saatuani onnistuneesti ajettua kaikki myyntijärjestelmän testit, VPN-tilausta lukuun ottamatta, läpi. Läpiajojen tarkoituksena oli tarkistaa, että kaikki testit toimivat odotetusti. Laajakaistan tilausprosessien kohdalla jouduin ajamaan testit läpi kaikilla eri modeemilaitevaihtoehdoilla niiden toimisen varmistamiseksi.

Eri tilausprosessien läpiajot eivät kuitenkaan sujuneet täysin ongelmitta. Esimerkiksi tilattaessa laajakaistaa kaapelitalouteen, jostain syystä Robotti saattoi välillä klikata väärää web-elementtiä tiettyä laajakaistapakettia haettaessa. Väärän elementin auetessa, Robotti ei pystynyt enää paikantamaan oikeaa elementtiä ja testi päätyi virheeseen. Tämä bugi ilmeni melko usein, mutta mitään suoranaista syytä tähän ongelmaan ei kuitenkaan löytynyt. Tämä johtui siitä, että väärää elementtiä, jota Robotti klikkasi virheen sattuessa, ei ollut määritelty itse koodiin. Itse epäilen virheen johtuneen siitä, että väärä elementti löytyi lähes samasta kohtaa kuin etsitty elementti ja bugin ilmetessä Robotti klikkasi sitä. Ongelman ratkaisemiseksi hidastin testin suoritusnopeutta. Välillä Robotti ei löydä etsittyä elementtiä, jos sivu ei ole ensin latautunut loppuun asti tai jos sivusto ei ole ehtinyt suorittamaan loppuun Robotin aikaisempaa toimintoa, esimerkiksi avaamaan alasvetovalikkoa ennen siitä löytyvän alkion hakemista. Säädettyäni testin suoritusnopeutta, testi toimi taas normaalisti.

Tiistai 8.8.2017

Tavoitteenani oli jatkaa asiakkaan omien sivujen testin ohjelmoimista, jossa Robotti vaihtaa valitun puhelinliittymän laskutustietoja. Tässä tapauksessa Robotin oli tarkoitus vaihtaa vain sähköpostiosoite, johon liittymän lasku lähetetään ja tämän jälkeen vielä tarkistaa, että laskutustietojen vaihto onnistui. Muiden yhteystietojen, kuten katuosoitteen, muuttaminen voi kollegoideni mukaan odottaa siihen asti jos tekemiäni omien sivujen testejä tarvitsee jatkossa päivittää tai kehittää.

Sain päivän aikana työstettyä hyvin kyseistä testiä eteenpäin, mutta en saanut sitä täysin valmiiksi. Automatisoimatta jäi vain muutama vaihe, kuten nykyisen laskutus sähköpostiosoitteen tallentaminen omaksi muuttujakseen, uuden sähköpostiosoitteen syöttäminen nykyisen osoitteen tilalle, muutoksen tallentaminen ja sen voimaantulon tarkistaminen. Samalla testiä ohjelmoidessani päivitin myös Xmind-tiedostoa, johon olin aikaisemmin kirjottanut kaikki verkkokaupan ja myyntijärjestelmän tilausprosessit vaiheineen ylös. Päivittäessäni omien sivujen prosessien kuvauksia, lisäsin muidenkin järjestelmien tilausprosesseihin Robotin koodiin päivityksien ja korjauksien yhteydessä kirjoittamani uudet vaiheet, joita ei dokumentoinnista vielä löytynyt.

Ongelmia päivän aikana tuottivat asiakkaan omien sivujen testiympäristön mahdolliset päivitys- tai muutostyöt, joiden takia sivuston aukeaminen kesti muutamasta minuutista lähes jopa kymmeneen minuuttiin. Tästä johtuen tekemieni koodien testaaminen oli hyvin hidasta, koska sivuston oli ensin auettava ennen kuin Robotti pystyi ajamaan tietyt testitapaukset. Näin ollen en kyennyt päivän aikana työstämään uutta testiä niin paljoa kuin olisin toivonut. Myös testiympäristöön liittyvien ongelmien selvittäminen ja niistä eteenpäin raportointi hidastivat työntekovauhtia entuudestaan.

Päivän oppiminen tapahtui vahvasti kertauksen kautta, käyttämieni avainsanojen ja toimintatapojen ollessa jo entuudestaan tuttuja. Vaikka en tämänkään päivän aikana oppinut täysin uusia asioita, se ei kuitenkaan haitannut. Mielestäni jo opittujen asioiden kertaus on kuitenkin lähes yhtä tärkeää, ellei jopa tärkeämpää kuin uuden oppimisen, koska kertauksen kautta taidot ja toimintatavat vain vakiintuvat entuudestaan.

Keskiviikko 9.8.2017

Tämän päivän tavoitteena oli omien sivujen testin viimeistely, jossa Robotti muuttaa käyttäjän mobiililiittymän laskutusosoitetta. Varsinaisen testin ollessa kokonaisuutena melko lyhyt verrattuna esimerkiksi verkkokaupan ja myyntijärjestelmän testeihin, saavutin päivän tavoitteen ja sain testin valmiiksi. Lisäsin testin loppuun myös ylimääräisen testitapauk-

sen, joka palauttaa muokatun mobiililiittymän alkuperäisen sähköpostiosoitteen uuden sähköpostiosoitteen tilalle. Ohjelmoin tämän ylimääräisen testitapauksen sen takia, jos järjestelmää testattaessa ei haluta tehdä pysyvää muutosta muokattavaan liittymään, vaikka kyseessä onkin testiympäristö ja testikäyttäjät.

Iltapäivällä järjestetyssä tilannekatsauksessa yritin demota kokouksen osallistujille automatisoimaani testiä, jossa Robotti tilaa verkkokaupasta uuden mobiililiittymän. Testi kuitenkin epäonnistui, koska sivustolla oli käynnissä päivitys- tai korjaustyöt ja niiden takia kaikki tarvittavat elementit, kuten tuotteen hintatiedot, eivät olleet sillä hetkellä näkyvissä eikä Robotti löytänyt tarvittavia elementtejä.

Tämä oli konkreettinen esimerkki regressiotestauksen toiminnasta ja tärkeydestä. Testattavaan järjestelmään tehdään muutos ja sen jälkeen testataan joko automatisoidusti tai manuaalisesti, että järjestelmä toimii korjauksista tai uusista ominaisuuksistaan huolimatta kuten sen pitäisikin. Tässä tilanteessa sivusto ei kuitenkaan toiminut odotetusti, koska kaikki hintatiedot eivät olleet normaalisti näkyvissä. Myöhemmin iltapäivällä ongelma oli korjattu ja verkkokauppaa testaavat testit toimivat taas normaalisti.

Torstai 10.8.2017

Saatuani eilen valmiiksi testin, jossa Robotti muuttaa testikäyttäjän mobiililiittymän laskutusosoitteen, tämän päivän tavoitteena oli aloittaa uusien asiakkaan omien sivujen testien ohjelmoiminen. Ennen uusien testien ohjelmoimista kävin vielä mobiililiittymän laskutusosoitteen muuttavan testin koodin läpi ja lisäsin selkeyden vuoksi ylimääräisiä lokikomentoja eli Log To Console-avainsanoja, joiden avulla Robotti kirjoittaa käytettävän kehitysympäristön, tässä tapauksessa PyCharmin, konsoliin halutun kommentin tai viestin (Robot Framework Foundation, 2017a). Näin pystyn entistä selkeämmin lukemaan lokista mitä vaihetta testistä Robotti suorittaa. Virheiden ja bugien etsimisen kannalta lokitus on hyvin hyödyllinen toimintapa, koska näin käyttäjä pystyy jo suurin piirtein PyCharmin lokista näkemään, missä mahdollinen virhe on tapahtunut ja koodin valmistuttua ylimääräiset lokimerkinnät on helppo poistaa.

Tämä toimintapa tuli itselleni tutuksi jo ensimmäisillä raportointiviikoilla ja käsittelenkin aihetta tarkemmin viikkoraportissa kaksi. Robot Frameworkia ohjelmoidessani olen oppinut toimimaan tämän toimintamallin mukaan sen käytännöllisyyden ja hyödyllisyyden takia. Koodin tarkistamisen ja testin läpiajojen yhteydessä päivitin myös Xmind dokumenttia saadakseni kaikki automaation suorittamat vaiheet ylös tekemäni dokumentaation myöhempää tarkastelua varten.

Suorittaessani testille muutamia läpiajoja sen toimivuuden tarkistamiseksi ja mahdollisten bugien löytämiseksi, törmäsin Robotin suorituksessa ongelmaan, joka on tullut myös aikaisemminkin vastaan muutamia verkkokaupan testejä ajattaessa. Tässä ongelmatilanteessa Robot Framework ilmoittaa, että XPath-polulla etsittävä web-elementti ei löydy sivulta tai että kyseinen elementti ei enää ole osa html-koodin rakennetta, DOM:ia vaikka elementti oli sivustolla selkeästi silmin havaittavissa. Kun html-dokumentti ladataan web-selaimeen, siitä tulee document object ja esimerkiksi kaikkien sivuston sisältämien elementtien ja attribuuttien ”juurielementti” (W3Schools, 2017a). Kyseisen ongelman ratkaisemiseksi kollegani oli työstänyt omaa, kustomoitua Robot Framework-kirjastoa, johon hän oli määritellyt valmiiden avainsanojen pohjalta uusia, tähän tilanteeseen suoraan soveltuvia avainsanoja. Kokeilemalla ja käyttämällä tämän kirjaston avainsanoja saimme ongelman ratkaistua ja korjasin aiempia koodejani sen verran lisää, että lisäsin kyseisen avainsanan myös niihin testeihin, joissa samankaltainen ongelma oli myös ilmennyt.

Testiympäristön teknisten ongelmien vuoksi en aivan heti päässyt aloittamaan uuden koodin kirjoittamista. Kyseinen testi lisää TV-sovellukseen halutun kanavapaketin. Saadakseni testin aloitettua pyysin käyttööni uusia testitunnuksia, joilla TV-palvelu olisi valmiiksi tilattuna, jotta uusien kanavapakettien lisääminen ylipäätänsä onnistuisi. Saatuani uuden testitunnuksen käyttööni pääsin viimein aloittamaan testin automatisoimisen. Olin aikaisemmilla viikoilla jo saanut kirjattua ylös tilausprosessin etenemisvaiheet mikä nopeutti jo entuudestaan lyhyen prosessin ohjelmoimista. En kuitenkaan vielä tämän päivän aikana saanut kyseistä prosessia kokonaan automatisoitua.

Perjantai 11.8.2017

Aloitettuani eilen kanavapakettien lisäämisen automatisoinnin tilattuun TV-palveluun, tämän päivän tavoitteena oli saada kyseinen testi automatisoitua niin pitkälle kuin mahdollista testitunnuksien ja -ympäristöjen rajoituksia silmällä pitäen. Tämän jälkeen tavoitteenani oli aloittaa viimeisten testien automatisointi eli olemassa olevan laajakaistatilauksen nopeuden ja tilaustietojen muuttaminen.

Sain kanavapakettitestille asettamani tavoitteet saavuttua eikä testistä puuttunut enää kuin tilausehtojen ja itse tilauksen vahvistaminen. Nämä puuttuvat vaiheet ovat myöhemmin ohjelmoitava sellaista testitunnusta hyödyntämällä, jolla TV-palvelu löytyy. Ohjelmoituani testin niin pitkälle kuin mahdollista, sain uuden testitunnuksen avulla myös aloitettua laajakaistan tilaustietojen muuttamisen ja sen nopeuden noston automatisoinnit. Pystyin

työstämään molempia automaatioita saman aikaisesti niiden ensimmäisten vaiheiden ollessa identtiset. Tämä nopeutti molempien testiautomaatioiden työstämistä.

Päivän aikana tapahtuneesta kertaamisesta huolimatta, suurin oppiminen tapahtui tietojeni karttuessa järjestelmien testi-instanssien välisistä integraatioista. Jouduin testiympäristöön liittyvien teknisten ongelmien takia selvittämään muutamien eri järjestelmien vastuuhenkilöiltä ja miten tieto, esimerkiksi tietyn tilauksen poistosta, kulkee eri järjestelmien testi-instanssien välillä. Näin oli mahdollista kollegoiden johdolla miettiä mistä mahdolliset tekniset virheet olisivat johtuneet. Integraatioihin liittyvä lisätieto oli kokonaisuuden hahmottamisen kannalta hyvin tärkeää ja se avasi myös uusia näkökulmia testaamiseen.

3.9.1 Viikkoanalyysi 9

Tämän viikon aikana oppiminen tapahtui taas pääsääntöisesti kertauksen kautta käyttäessäni aikaisemmin oppiminiani Robot Frameworkin ohjelmointiin tarvittavia taitoja sekä hyväksi näkemiäni toimintatapoja uusia testejä automatisoidessani. Tämän, kuten parin viimeisimmänkin viikon aikana, olen pystynyt hyvin käytännön kautta harjoittelemaan oppimieni sekä uusien että vanhojen avainsanojen käyttöä.

Hyödyntäessäni oppejani eri testeissä ja erilaisissa tilanteissa olen samalla pystynyt soveltamaan ja kokeilemaan eri tapoja käyttää aikaisemmin hyödyntämiäni avainsanoja. Näin olen pystynyt opettelemaan minkälaisiin tilanteisiin mitkäkin avainsanat soveltuvat parhaiten. Olen myös pystynyt uusia koodeja ohjelmoidessani ennakoimaan mitä kirjastoja ja niiden avainsanoja minun olisi paras hyödyntää uusissa testeissä, jotta ohjelmoiminen olisi mahdollisimman käytännönläheistä ja helppoa. Näin ohjelmointi pysyisi myös mahdollisimman mielekkäänä.

Tilanteissa, joissa en ole suoraan kokemuksieni perusteella tiennyt miten minun tulisi edetä, vastaus on löytynyt nopeasti eri lähteitä käyttämällä. Olen myös kollegoideni työskentelytapoja seuraamalla löytänyt uusia etenemismahdollisuuksia ja toimintatapoja Robotin ohjelmoimisen suhteen. Tällaisia toimintatapoja ovat muun muassa olleet ylimääräiset lokimerkintöjen ja muuttujien kirjoitustyylin vakiintuminen.

Keskiviikkona järjestetyssä demotilaisuudessa sain huomata kuinka hankalaa verkkosivujen automatisointi voi olla. Kyseisessä tilaisuudessa esittämäni demo epäonnistui siksi, että testattavalta verkkosivulta puuttui päivitystöiden takia tarvittavia elementtejä, joita Robot Frameworkin oli tarkoitus etsiä. Juuri nämä käyttöliittymän jatkuvat muutokset ovat suuri ongelma graafisen käyttöliittymän automaattisessa testaamisessa sivuston muuttu-

essa uusien versioiden ja päivitysten myötä (Kaner, Bach & Pettichord, s.106). Tämä tulee ongelmaksi myös siinä, että haluttua sivun versiota ei voi vain jäädyttää ja olla muuttamatta. Muutostöiden ohella graafisen käyttöliittymän automaattisessa testaamisessa ongelmana voi olla myös se, että käyttöliittymä saattaa hajota tai lakata toimimasta useinkin (Page, s.12) hankaloittaen sen automaattista testaamista.

Normaalisti niin regressiotestauksessa kuin muunlaisessakin testauksessa kehitys- ja testaustiimien tulisi olla hyvin aktiivisessa vuorovaikutuksessa keskenään, jotta tiimit pystyvät varmistamaan, että informaatio varmasti kulkee eri osapuolien välillä. Nyt itselläni ei ollut mitään tietoa siitä, että testaamani sivustoa, jota minun oli tarkoitus demota, työstettiin juuri samaan aikaan. Tämä saattoi mahdollisesti johtua siitä, että olen vasta automatisoimassa testejä, joiden avulla regressiotestausta suoritettaisiin. Olisi silti oletettavaa ja hyödyllistä, että tieto päivitys- tai muutostöistä kulkisi vähintään tiimitasolle. Näin tieto mahdollisesti ilmenevistä ongelmista olisi läsnä eikä testaajille tuli yllätyksiä esimerkiksi juuri esittelytilaisuuksissa.

Viikon aikana pääsin kollegani johdolla myös viimein tutustumaan Robot Frameworkin kustomoituihin kirjastoihin, ja näitä muita kuin Robot Frameworkin standardikirjastoja kutsutaan ulkoisiksi, external, kirjastoiksi (Robot Framework Foundation, 2017c). Vaikka kyseessä olikin hyvin nopea katsaus näiden kirjastojen toimintaan, opin silti uutta liittyen kustomoituihin kirjastoihin. Lyhyen katselmuksen aikana sain jo melko selkeän kuvan kustomoitujen kirjastojen rakenteesta ja toiminnasta.

Kuten valmiitkin Robot Frameworkin kirjastot, kustomoidut kirjastot koostuvat käyttäjän itse määrittelemistä avainsanoista, joiden toiminta on ohjelmoitu Python-ohjelmointikielellä. Pelkän Pythonin lisäksi avainsanojen toiminnallisuuksien määrittämisessä voidaan kollegani mukaan hyödyntää myös esimerkiksi erilaisia api-rajapintoja tai muita valmiita kirjastoja. Omia avainsanoja voidaan luoda täysin alusta alkaen tai hyödyntämällä valmiiden kirjastojen avainsanojen toiminnallisuuksien takaa löytyvää koodia työskentelyn pohjana. Käyttäjä voi hyödyntää valmiin avainsanan koodia oman avainsanansa pohjana esimerkiksi sellaisessa tilanteessa, jos käytetty avainsana on muuten sopiva vastaan tulleeseen tilanteeseen, mutta siitä puuttuu tarpeellinen ominaisuus tai toiminto.

Itse näen asian niin, että yleisesti ottaen kustomoituja kirjastoja otetaan käyttöön siinä vaiheessa, kun Robot Frameworkin valmiit kirjastot avainsanoineen eivät riitä tiettyjen toiminnallisuuksien suorittamiseen. Tämä voi ilmetä myös esimerkiksi niin, että Robotin suorittama testi päätyy usein tiettyä avainsanaa suoritettaessa virheeseen. Testin päättyminen usein samaan virheilmoitukseen, samassa kohdassa koodia toi minulle tällä viikolla

mahdollisuuden perehtyä tarkemmin kustomoituihin kirjastoihin. Olin kahdeksannella raportointiviikolla jo pääsyt hieman näkemään tällaisten kirjastojen rakennetta, mutta tällä viikolla otin käyttööni yhden kollegani tekemistä kirjastoista toistuvan virheilmoituksen takia.

Ajaessani torstaina läpi asiakkaan omien sivujen testiä, jossa mobiililiittymän laskutusosoite muutetaan, Robotti ei löytänyt sivulta löytyvää web-elementtiä ja ilmoitti, että kyseinen elementti ei myöskään löytyisi lainkaan sivuston koodista vaikka elementti oli silmin havaittavissa. Käytin tässä tilanteessa Wait Until Element Is Visible-avainsanaa, joka odottaa määritellyn ajan, että sijainnin, esimerkiksi koordinaattien tai XPath-polun avulla määritelty elementti on näkyvissä (Robot Framework Foundation, 2017d). Avainsanan tarkistaessa onko etsitty elementti sivulla näkyvissä, se havaitsikin virheellisesti, että koko elementtiä ei olisi sivuston koodissa ollenkaan ja tästä syystä testi päätyi virheeseen. Tämän vuoksi otimme käyttöön kollegani itse määrittämän avainsanan, joka myös varmistaa, että elementti on näkyvissä DOMissa. Tämän avainsanan avulla kyseinen, useasti ilmenevä virhe saatiin karsittua pois.

Kuluneen viikon aikana jouduin selvittämään paljon asioita liittyen käyttämäni testiympäristön järjestelmien testi-instanssien välisiin integraatioihin. Tämä johtui jumiin jääneestä VPN-palvelun tilauksesta, joka häiritsi tilausprosessin suorittamista jo viikoilla 7 ja 8. Tarkoituksenani oli saada tarkempaa tietoa siitä, miten esimerkiksi tieto tilauksen poistosta kulkee testiympäristössä eri järjestelmien testi-instanssien välillä. Selvittelytyö oli melko työlästä, koska ensin oli selvitettävä kuka mistäkin järjestelmästä vastaa ja tämän jälkeen asiaa oli lähestyttävä vastuuhenkilö kerrallaan.

Kuten olin aikaisemmin jo julkiselta sektorilta saamieni kokemuksieni kautta huomannut, kommunikointi eri järjestelmien vastuuhenkilöiden välillä on hyvin tärkeää erilaisten ongelmien ratkaisemiseksi. Mielestäni, varsinkin testausta silmällä pidettäessä, kommunikoinnin aktiivisuuteen voisi parantaa myös järjestelmien vastuuhenkilöiden suunnalta. Näin testaajat saisivat aina tarvittavat tiedot esimerkiksi sivustojen ja järjestelmien tulevista päivitystöistä ja mahdollisista virhetiloista eikä tieto saavuttaisi testaajia vasta siinä vaiheessa kun testattava järjestelmä ei enää toimi kuten aikaisemmin. Testaajille olisi hyvä myös mahdollisesti olla saatavilla mahdollisimman paljon yleistä tietoa järjestelmään ja sen ominaisuuksien liittyen työn helpottamiseksi ja taas erilaisten virhetilojen välttämiseksi.

Uusien ongelmien ja niistä johtuvien selvitystöiden takia uusia sidosryhmiä nousee jatkuvasti esille. Tämä helpottaa asioiden selvittämistä jatkoa silmällä pitäen, koska uusien

ongelmien ilmetessä asiasta tietävä henkilö on jo selvillä, jolloin häntä voi lähestyä asiaan liittyen suoraan. Jossain tapauksissa, kuten Kaupunginkansliassa työharjoitteluni aikana sain huomata, tieto siitä, kuka järjestelmän vastuuhenkilö on, ei ole aina sitä tarvitsevien henkilöiden tiedossa. Välillä vastuu tietystä järjestelmästä ei välttämättä ole edes itse vastuuhenkilön tiedossa. Näissä tapauksissa vastaan tulleen ongelman suuntaaminen oikealle henkilölle voi olla hyvin vaikeaa ja selvitystyö vie taas lisää aikaa varsinaisesta työstä.

Onneksi tällaiset tilanteet ovat loppujen lopuksi hyvin harvassa. Yleisesti ottaen testaajilla tulisi myös olla hyvin selvillä eri järjestelmien vastuuhenkilöt varsinkin erilaisten ongelmatilanteiden varalta, jotta testien, tässä tapauksessa testiautomaatioiden, suorittaminen ei kärsi tiedon puutteen takia mikä pätee myös yleistietoon testattavista järjestelmistä sekä aktiiviseen kommunikointiin eri osapuolten välillä.

3.10 Seurantaviikko 10

Maanantai 14.8.2017

Päivän tavoitteena oli saada viimeisteltyä asiakkaan omien sivujen testi, jossa Robotti muuttaa testikäyttäjän laajakaistan laskutustiedot. Päivälle asettamani tavoite oli melko helposti saavutettavissa, koska testin automatisoinnissa ei ollut enää montaa vaihetta ohjelmoimatta. Testiä työstäessäni pystyin myös hyödyntämään aikaisemmin ohjelmoimaani omien sivujen testiä, jossa Robotti muuttaa testikäyttäjän mobiililiittymän tiedot.

Sain saavutettua päivälle asettamani tavoitteen ja ohjelmoitua testin lopputestausta vaille valmiiksi. Testi oli siis ohjelmoitu, mutta en saanut vielä testattua viimeiseksi lisäämieni testitapausten toimivuutta käytännössä. Teoriassa viimeisten testitapausten koodin pitäisi toimia, koska ne ovat muuttujien nimiä ja arvoja lukuun ottamatta täysin identtinen mobiililiittymän tietojen muuttamisen viimeisten vaiheiden testitapausten kanssa. Myös sivulta löytyvät elementit ovat saman nimisiä.

Iltapäivästä sain vihdoin ohjeet siihen, miten saisin aloitettu puuttuvan myyntijärjestelmän testin ohjelmoimista, jota en aikaisemmilla viikoilla pystynyt aloittamaan. Kyseisestä testistä oli puhetta jo aikaisemmilla viikoilla, mutta en ollut saanut selkeitä ohjeita sen automatisoimisesta. Tässä testissä asiakkaalle tilataan hänen ilmoittamaansa laitteeseen laitevakuutus mahdollisten laitteeseen kohdistuvien vahinkojen varalle. Testiä oli helppo lähteä toteuttamaan, koska kyseessä oli turvapaketin ja VPN-yhteyden tavoin VAS-tuotteen tilausprosessi, joten pystyin hyödyntämään lähes täysin jo valmista koodia. Tämä helpotti ja nopeutti huomattavasti työn etenemistä.

Päivän aikana en varsinaisesti oppinut uusia asioita Robot Frameworkiin liittyen vaan pystyin suoraan hyödyntämään jo oppimiani ja oman tasoni mukaisesti hallitsemiani avainsanoja. Kuten muissakin viimeksi ohjelmoimissani testeissä, oppiminen tapahtui nytkin jatkuvasti kerratessani ja käyttäessäni uudestaan oppimiani taitoja. Uutta oppimista on pääosin tapahtunut silloin, kun olen erilaisten ongelmanratkaisutilanteiden kautta löytänyt niin uusista kuin vanhoista kirjastoista ja avainsanoista uusia käyttötapoja, joita en ole ollut aikaisemmin huomannut tai osannut hyödyntää.

Tiistai 15.8.2017

Aloitettuani eilen uuden myyntijärjestelmän testin automatisoinnin, tämän päivän tavoitteena oli saada se viimeisteltyä niin pitkälle kuin mahdollista. Työn jatkamista helpotti huomattavasti se, että pystyin uuden testin automatisoinnissa hyödyntämään valmiita testitapauksia. Valmiin koodin lisäksi minun oli kuitenkin ohjelmoitava muutama uusi testitapaus, koska kaikki VAS-tuotteiden tilaamiseen tarkoitetut testitapaukset eivät suoraan taipuneet uuden tilattavan tuotteen tietojen tarkasteluun. Näiden uusien testitapauksien ohjelmoiminen sujui lähes ongelmitta hyödyntäessäni aikaisemmin tekemääni koodia mallina.

Saavutin päivälle asettamani tavoitteen saadessani uuden testin automatisoinnin ohjelmoitua niin pitkälle kuin testiympäristön toiminnan puitteissa oli mahdollista. Viime viikosta asti läsnä olleen teknisen vian takia testin toimivuutta ei voinut täysin testata. Saavutettuani päivän tavoitteen jatkoin asiakkaan omien sivujen testien ohjelmointia ja sain kanavapakettien tilaamista koskevan automaation tilauksen hyväksymistä vaille valmiiksi.

Päivän aikana opin uutta Javascript-ohjelmointikielen käytöstä, jonka komentoja pystyy suoraan Robot Frameworkin kautta ajamaan Execute Javascript-avainsanan avulla (Robot Framework Foundation, 2017d). Olin jo aikaisemmilla raportointiviikoilla hyödyntänyt kyseistä avainsanaa ja erilaisia Javascript-komentoja eri testeissäni sen hyödyllisyyden ja toimivuuden takia. Tänään opin vihdoin palauttamaan Javascript-komentojen avulla esimerkiksi halutun elementin arvon ja asettamaan sen muuttujan arvoksi. Tämän avulla pystyin esimerkiksi ensin tarkistamaan Javascriptin avulla, onko tietty checkbox- tai radiopainike aktivoitu ja saadun tuloksen (tosi tai epätosi) perusteella pystyin suorittamaan esimerkiksi halutun ehtolauseen.

Keskiviikko 16.8.2017

Keskiviikon tavoitteena on saada eri asiakkaan omien sivujen testien vieminen niin pitkälle kuin mahdollista. Päädyin työstämään omien sivujen testeistä laajakaistan laskutusosoitteen muuttamista, minkä sain myös työstettyä täysin valmiiksi.

Päivän tavoitteen saavuttamisen jälkeen päätin oppimisen kannalta kokeilla uusia, erilaisia vaihtoehtoja, joilla saisin eilen valmiiksi saamani myyntijärjestelmän laitevakuutuksen automatisoinnin toimimaan paremmin. Tarkoituksena oli löytää koodin hallinnoinnin kannalta paras mahdollinen ratkaisu testitapaukseen, joka tarkistaa laitevakuutukseen liittyvät hintatiedot. Nykyinen ratkaisu oli mielestäni hieman vaikeasti seurattava, varsinkin ulkopuolisia lukijoita silmällä pitäen. Testattuani useita eri tapoja ohjelmoida kyseinen testitapaus, päädyin lopulta jakamaan sen selkeyden takia useaan lyhempään testitapaukseen. Päivän aikana opin taas lisää eri järjestelmien välisestä tietoliikenteestä käyttämässämme testiympäristössä

Päivän ongelmaksi nousi koodia uudelleen järjestellessä Robot Frameworkin logiikasta löytynyt bugi. Bugi koski käyttämääni Run Keyword If-avainsanaa, jossa ennalta määritelty toiminto suoritetaan vain, jos haluttu ehto toteutuu (Robot Framework Foundation, 2017a). Bugi ilmeni kun Robotti tyhjensi suoraan ehtolauseessa käsitellyn muuttujan arvon, jos määritelty ehto ei toteutunut. Tätä kuitenkaan ei olisi pitänyt tapahtua, vaan muuttujan arvon olisi pitänyt pysyä muuttumattomana. Ilmoitettuani löytämästäni bugista kollegalleni, pystyin kiertämään kyseisen bugin jakamalla kirjoittamani testitapauksen useampaan eri testitapaukseen yhden monimutkaisen ehtolauseen sijasta ja näin käsitellyn muuttujan arvokaan ei tyhjentynyt.

Torstai 17.8.2017

Saatuani haltuuni uusia testitunnuksia, asetin tämän päivän tavoitteeksi ohjelmoida laajakaistayhteyden nopeuden nostamisen ja kanavapakettien lisäämisen loppuun. En saanut päivän tavoitetta valitettavasti saavutettua, koska kanavapakettien lisäämisen automatisointi olikin työläämpää kuin oletin. Tämän takia en saanut kumpaakaan testeistä valmiiksi. Kanavapakettien lisäämisestä jäi puuttumaan enää tilauksen hyväksyminen ja sen käynnissä olemisen tarkistaminen. Laajakaistanopeuden nostamisen automatisointia en päässyt aloittamaan ollenkaan.

Ongelmaksi ja haasteeksi kanavapaketin tilausta automatisoidessa nousi sellaisen testitapauksen ohjelmointi, joka valitsee sivustolta sellainen tilauksen, johon on mahdollista tilata uusia kanavapaketti. Tämä tarkoitti sitä, että Robotti oli ohjelmoitava tarkistamaan, että valittava tilaus ei ole käsittelyssä. Robotin oli samalla myös varmistettava myös, että

jos samaan osoitteeseen oli kaksi samannimistä tilausta, Robotti valitse niistä oli ensimmäisen vapaana olevan.

Perjantai 18.8.2017

Koska en saavuttanut eiliselle päivälle asettamiani tavoitteita, tämän päivän tavoitteena oli viimeistellä ainakin kesken jäänyt testi kanavapakettien lisäämiseksi. Tavoitteena oli myös, jos aikaa jäisi, aloittaa viimeisen omien sivujen testin automatisointi, jossa Robotti päivittää valitun laajakaistayhteyden nopeuden.

Saavutin päivälle asettamani päätavoitteen ja sain viimeisteltyä kanavapakettien tilaamisen automasinoinnin ongelmitta. Samalla tein tarvittavat testiajot testin toimivuuden varmistamiseksi, varmistin myös, että muut kesätyösuhteeni aikana ohjelmoimani testit toimisivat. Näitä testejä ajaessani päätin lisätä jo valmiiseen testiin, joka muuttaa valitun mobiililiittymän laskutustietoja, vielä uuden testitapauksen. Tämä testitapaus tarkistaa, että jos liittymän tietojen muutokset eivät ole heti näkyvissä, Robotti lataa sivun uudelleen. Päätin lisätä kyseiseen testiin tämän testitapauksen, koska huomasin ajaessani kyseistä testiä, että muutettujen tietojen päivittymisen näkyminen saattoi vaatia sivun uudelleen lataamista.

Saavutin myös päivän toisen tavoitteeni ja ehdin iltapäivän aikana aloittamaan laajakaistan nopeuden muuttamisen automatisointia. Viimeisten kymmenen viikon aikana oppimani avulla, kyseisen testin automatisointi oli melko helppoa ja nopeaa. Hankaluuksia kuitenkin tuotti eritoten tilausvahvistuksen tarkistamisen automatisointi, koska tilausvahvistussivun käyttöliittymän ulkoasu vaihteli riippuen siitä, onko nykyisen laajakaistan asentamisen yhteydessä asennettu myös TV-palvelu. Tämä hidasti automaation loppuun viemistä, varsinkin kun syy käyttöliittymän muuttumiseen ei ollut aluksi selvillä. Ohjelmoitavaa ei kuitenkaan jäänyt enää paljoa.

Seuraavan ja viimeisen työviikon tavoitteeksi minulle jää vielä työstää ja viimeistellä loput omien sivujen testit. Samalla minun täytyy viimeistellä kesken oleva verkkokaupan testi, jossa käyttäjä tilaa uuden mobiililaiteen kuukausimaksulla. Tämän testin viimeistelemiseksi tarvitsen kuitenkin vielä ylimääräiset testitunnukset, joiden luottotiedot ovat kunnossa. Työstettyäni loput puuttuvat ja keskeneräiset testit minulla pitäisi olla loppuviikosta aikaa hioa kaikkia testejä tarvittaessa vielä lisää ja tarkistaa niiden toimivuus.

3.10.1 Viikkoanalyysi 10

Kymmenennen ja viimeisen raportointiviikon aikana osaamiseni Robot Frameworkin parissa kehittyi askeleen eteenpäin aikaisempien viikkojen tapaan. Tämän viikon aikana en kuitenkaan varsinaisesti oppinut kuin muutamia täysin uusia yksittäisiä asioita ja tällaisia asioita olivat esimerkiksi Javascript-komennolla haetun arvon tallentaminen muuttujaan. Muuten hyödynsin ja sovelsin viimeisten 10 raportointiviikon aikana eri lähteistä oppimaani teoriaa, käytäntöjä sekä toimintatapoja.

Keskiviikkona ongelmaksi nousi Robot Frameworkin logiikasta löytämäni bugi. Tästä bugista johtuen Robotti tyhjensi ehtolauseessa käsitellyn muuttujan arvon, jos muuttuja koskeva ehto ei toteutunut. Näin ei kuitenkaan olisi pitänyt käydä, vaan muuttujan arvon oli pitänyt pysyä muuttumattomana. Tähän mennessä en ollut hirveästi miettinyt sitä mahdollisuutta, että Robot Frameworkista löytyisi tällaista virhettä. Tämä oli virheellinen kuvitelma, koska harvemmin mikään sovellus tai työkalu on täysin, 100 prosenttisesti bugivapaa. Kanerin, Bachin & Pettichordin (2002, s.104) mukaan testityökalut, varsikin maksulliset, saattavatkin sisältää enemmän bugeja kuin verrattavissa olevat kehitystyökalut ja nämä bugit hankaloittavatkin syiden kartoittamista automatisoitujen testien epäonnistumisien takana.

Tämän sain huomata tämän viikon aikana itsekin. Virheen sattuessa en osannut itse aluksi edes sanoa, johtuiko testin epäonnistuminen omista ohjelmointitaidoistani vai itse Robot Frameworkista. Mahdollisista bugeista johtuen on hyvä, että Robotin lähdekoodi on kaikille avointa ja kaikilla on mahdollisuus käyttää sitä. Näin käyttäjät pystyvät ilmoittamaan Robot Frameworkin kehittäjille löytämistään kyseisen viitekehyksen sisältämistä virheistä ja kehittäjät pystyvät käyttäjäyhteisön kanssa parantamaan tuotettaan entuudestaan.

Viikon aikana vastaani tuli myös lisää selvitettävää ja ongelmia liittyen käytettävään testiympäristöön ja järjestelmiin. Myös testitunnuksiin liittyen esiin nousi lisää ongelmia, kun osa testitunnuksista ei sisältänyt tarvittavia ominaisuuksia ja osa testitunnuksista ei toiminut ollenkaan esimerkiksi asiakkaan omien sivujen kohdalla. Selvitettyäni asiaa, osa näihin tunnuksiin liittyvistä ongelmista johtui siitä, että samalla yksilöivällä testitunnuksella saattoi löytyä käytetystä testiympäristöstä kaksi käyttäjää yhden sijasta.

Testitunnuksiin ja – ympäristöihin liittyvät ongelmat selvisivät kun otin yhteyttä niihin henkilöihin, joilla oli tarvittava tietotaito kyseisistä aiheista. Heiltä sain esimerkiksi tarvittavat ominaisuudet omaavat uudet testitunnukset ja heidän kauttaan sain myös samalla selitykset käytettävää testiympäristöä ja sen testijärjestelmiä koskeviin kysymyksiin. Näin osaa-

miseni myös eri järjestelmien testi-instansseihin liittyen kasvoi. Kuten esimerkiksi yhdeksännellä raportointiviikolla, tämä helpotti järjestelmien välisen toiminnan kokonaiskuvan hahmottamista. Sain taas parempaa käsitystä siitä mikä järjestelmä on miten toiseen järjestelmään yhteydessä ja ovatko tietyt vastaan tulleet ongelmat voineet johtua esimerkiksi testidatan eheydestä tai puutteista järjestelmien välisessä tiedon kulussa.

Olen varsinkin tämän työsuhteen ja etenkin kuluneen viikon aikana oppinut tavoitteiden asettamisesta sen, että vaikka tietylle ajanjaksolle asetetut tavoitteet, tässä tapauksessa päivälle, olisivatkin saavutettavissa, ne eivät välttämättä toteutuisivatkaan. Tämä on ollut itselleni itsestään selvää jo aikaisempien työkokemuksieni kautta, mutta varsinkin muutamien kuluneiden viikkojen aikana se on ollut hyvä huomata konkreettisesti uudestaan. Esimerkiksi tällä viikolla useat testit, joiden piti olla yhden päivän aikana tehtävissä, osoittautuivatkin odotettua työläämmiksi esimerkiksi muuttuvan käyttöliittymän takia.

Samalla tavalla yhtä testiä työstettäessä, toisen testin koodista saattoi löytyä korjattavaa ja virheitä, joiden selvittäminen vei myös aikaa. Pyrin useita virheitä löytäessäni keskittymään yhteen asiaan kerralla, jotta töiden hallinnointi pysyi mahdollisimman hyvin kasassa eikä ajatus päässyt katkeamaan keskittymisen pysyessä yhdessä asiassa kerrallaan. Tietysti Robot Frameworkia ohjelmoidessa vastaan saattoi tulla myös testitapauksia, joita käytettään useassa eri testissä, jolloin testitapauksien toimivuus kaikissa niitä käyttävissä testeissä oli varmistettava. Työtehtävien laajuuden lisäksi päivän aikana vastaan saattoi tulla myös itsestään riippumattomia ongelmia, kuten järjestelmiin tai testattaviin ympäristöihin liittyviä ongelmia, joista johtuen päivälle asetetut tavoitteet eivät aina toteutuneet.

Huomatessani tämän työsuhteen aikana taas konkreettisesti, että päivälle asetetut tavoitteet eivät välttämättä toteutuisi vaikka kuinka motivoituneesti tahansa työskentelisi, vakiinnutin itselleni käytännön asettaa eritasoisia tavoitteita ja välitavoitteita. Näin ollen päivälle ei tule asetettua liian suuria tavoitteita. Varsinkin testejä ohjelmoidessani huomasin, että tavoitteita oli helpompi asettaa ohjelmoitavien testien laajuuden ja vaikeuden selvitessä päivän aikana.

Testejä automatisoidessani pyrin esimerkiksi ensin ottamaan tavoitteeksi tietyn testitapauksen ohjelmoimisen, jonka jälkeen oli helppo asettaa päivän seuraavaksi tavoitteeksi seuraavan testitapauksen ohjelmoiminen ja niin edelleen. Näin päivän päätavoitteena saattoi hyvin olla tietyn testin työstäminen ja tämä päätavoite koostui pienemmistä välitavoitteista. Välitavoitteiden asettaminen helpotti huomattavasti myös työn jaksottamista ja pyrin asettamaan niitä aktiivisemmin jatkossa myös niin opiskeluissani kuin myös tulevissa työsuhteissani.

4 Pohdinta ja päätelmät

Tutkiessani kymmenen raportointiviikon jälkeen ensimmäisellä työviikolla kirjoittamaani nykytilanteen kuvausta, huomaan, että omat taitoni varsinkin Robot Frameworkin ohjelmoimiseen liittyen ovat kehittyneet selkeästi. Kuten nykytilanteen kuvauksessa kuvataankin, lähes kaikki tämän kesätyösuhteen aikana käsittelemäni aiheet olivat minulle täysin entuudestaan tuntemattomia. En ollut ennen opinnäytetyöni aloittamista kuullut Robot Frameworkista ja kaikki testaamiseen liittyvä kokemukseni perustui manuaalisen testauksen kautta saatuun kokemukseen. En myöskään ollut aikaisemmin käyttänyt tehtävänhallintaohjelmisto Jiraa ja sähköposti- ja pikaviestintäsovelluksia lukuun ottamatta mitään työtehtävissäni tarvitsemiani järjestelmiä.

Nyt kymmenen raportointiviikon ja kahdentoista työviikon jälkeen, pystyn itsenäisesti luomaan omia Robot Framework testejä ja hyödyntämään eri Robot Frameworkin valmiskirjastoja ja niiden sisältämiä avainsanoja. Ymmärrän myös, miten testitapaukset ja avainsanat muodostavat omia testikokonaisuuksiaan. Työsuhteeni aikana opin myös määrittelemään monimutkaisiakin XPath-polkuja erilaisten web-elementtien paikantamiseksi Robotilla. Myös muiden itselleni uusien työkalujen, kuten Jenkinsin, Xmindin, Deveon ja Git-Bashin, käyttö tuli aikaisempaa tutuksi.

Alun perin työtehtävien määrittyessä, puheena oli, että pyrkisin automatisoimaan niin monta testiä kun vain annetun ajan ja omien taitojeni puitteissa olisi mahdollista. Tämä johtui osittain siitä, että testiautomaatio ja Robot Framework olivat minulle entuudestaan melko tuntemattomia aihealueita. Tehokkaan koulutuksen, motivoituneen ja aktiivisen itsenäisen työskentelyn, saatavilla olevien materiaalien ja avun avulla pystyin kuitenkin suorittamaan kaikki minulle annetut työtehtävät. Vaikka viimeisten omien sivujen ja verkkokaupan testien työstäminen venähti viimeiselle työviikolle, muun muassa puutteellisten testitunnuksien takia, sain silti tehtäväni suoritettua ajallaan. Kyseessä oli siis kolmea eri järjestelmää testaavien 13 eri testin automatisointi onnistuneesti.

Vaikka jokaisella työ- ja raportointiviikolla tapahtui huomattavaa kehitystä, annetuissa työtehtävissä ansiokkaasti suoriutuminen uusia työkaluja ja järjestelmiä käyttäen oli itselleni suurin merkki omasta kehityksestäni. Näin ollen saavutin tavoitteeni ja pystyn huomaamaan kesätyösuhteen aikana tapahtunutta ammatillista kehitystäni yleisesti ja etenkin testiautomaatioon liittyen.

Työskennellessäni Robot Frameworkin parissa koko kesätyösuhteeni ajan, opin paljon uusia ratkaisumalleja kyseiseen viitekehykseen liittyen. Hyödyntäessäni mahdollisimman monipuolisesti eri lähteitä Robot Frameworkin ohjelmoinnissa, pystyin kokeilemaan useita erilaisia ratkaisumalleja ja toimintatapoja ja valitsemaan niistä ne itselleni sopivimmat. Tämä johtui siitä, että Robottiin liittyen löytyy hyvin paljon materiaalia sen lähdekoodin ollessa avointa. Tämä oli hyödyllistä varsinkin erilaisissa ongelmatilanteissa kun erilaisia ratkaisumalleja oli saatavilla useita erilaisia. Tärkeintä tässä ratkaisutavassa oli lähteä rohkeasti kokeilemaan erilaisia ratkaisumalleja, vain tuttujen ja turvallisten ratkaisumallien sijasta.

Loppujen lopuksi eniten ratkaisumalleja sain käyttööni testiautomaatiosta vastaavalta kollegaltani. Hänen kouluttaessaan minulle miten Robot Framework toimii, opin häneltä suoraan tietyn tavan kirjoittaa Robotin koodia. Näitä sisäistämiäni ratkaisumalleja pystyin pitämään pohjana omalle toiminnalleni. Yksi hyvin hyödyllinen ratkaisumalli oli koodin laaja kommentointi ja ylimääräisten lokimerkintöjen käyttö. Tämä tapa oli minulle jo entuudestaan tuttu ammattikorkeakoulun ohjelmointikursseilta, mutta tämän työsuhteen aikana sen käyttö vakiintui entuudestaan omassa toiminnassani. Tällaisten ylimääräisten lokimerkintöjen avulla pystytään helpommin ja nopeammin seuraamaan missä mahdollinen virhe on tapahtunut ja missä vaiheessa koodia testi etenee. Tämä ratkaisumalli on myös siinä mielessä hyvä, että se toimii ohjelmointikielestä riippumatta.

Toinen yleishyödyllinen ratkaisumalli oli testitapausten selkeä nimeäminen ja koodin pitäminen mahdollisimman selkeänä ja yhtenäisenä. Tämän ratkaisumallin ja selkeän kommentoinnin myötä koodista pystyy suoraan näkemään, mitä mikäkin testitapaus tekee mikä helpottaa omaa työtä huomattavasti. Näin myös ne ulkopuoliset lukijat, joille käytettävä ohjelmointikieli ei ole tuttu, pystyvät seuraamaan mitä koodissa milläkin rivillä tapahtuu.

Varsinaisiin työelämän käytäntöihin liittyviä ratkaisumalleja en tämän työsuhteen aikana oppinut suoranaisesti lisää. Yleiseen toimintaan ja kommunikointiin liittyvät ratkaisumallit olivat vakiintuneet aikaisempien työsuhteideni aikana pisteeseen, johon olin jo hyvin tyytyväinen. Tämän työsuhteen aikana sain huomata näiden ratkaisu- ja toimintamallien toimivuuden käytännössä erilaisissa vuorovaikutus- ja ongelmatilanteessa.

Yhtenä hyödyllisenä menetelmänä pyrin oman kehitykseni ja oppimiseni maksimoimiseksi jatkuvasti etsimään ja kokeilemaan uusia ja käytännöllisempiä tapoja automatisoida testejä. Pyrin samalla myös tarkastelemaan koodiani mahdollisimman kriittisestä näkökulmasta parantaakseni tekemieni testien ja testitapauksien rakennetta ja tehdäkseni koodista

mahdollisimman selkeälukuista. Pitäessäni kollegani kanssa erimittaisia koodikatselmuksia, huomasin sen olevan myös erinomainen menetelmä oman koodin laadun parantamiseksi ja uuden oppimiseksi. Ulkopuolisen tutkiessa koodia, siitä voi helposti löytyä huolimattomuusvirheitä tai muuta parannettavaa, mitä kirjoittaja ei ole itse pystynyt huomaamaan. Näiden eri menetelmien toimivuuden sain huomata käytännön kautta eri raportointiviikkojen aikana ja tarkastellessani omia aikaansaannoksiani.

Tämän päiväkirjamuotoisen opinnäytetyön aikana opin ohjelmoimaan automaattisia testejä Robot Frameworkilla ja käyttämään työtehtävissäni vaadittuja järjestelmiä ja työkaluja. Yksi näistä oli yhtenä tärkeimmistä työkaluistani toiminut PyCharm-kehitysympäristö. Oppiminen oli selkeästi havaittavissa, koska ennen kesätyösuhteeni alkua kyseisiin aihealueisiin liittyvä osaamiseni oli lähes olematon. Opin työssäni käyttämään eri Robot Frameworkin valmiskirjastoja ja niiden sisältämiä avainsanoja. Opin myös määrittelemään erilaisia XPath-polkuja erilaisten web-elementtien paikantamiseksi. Työsuhteeni aikana opin paljon myös muista itselleni uusista työkaluista ja järjestelmistä, kuten Jenkinsistä, Xmindista, Deveosta ja GitBashista.

12 työviikon aikana opin myös työskentelemään uudessa työympäristössä ja suhteuttamaan oman toimintani tämän ympäristön mukaan. Sain myös vakiinnutettua tapaani tavoitteiden saavuttamisessa asettamalla työpäivilleni selkeitä pää- ja välitavoitteita. Työskennellessäni ja taitojeni karttuessa, opin myös haastamaan itseäni entistä enemmän tarkastelemalla jatkuvasti omaa toimintaani ja aikaansaannoksiani.

Tämän opinnäytetyön aikana huomasin, kuinka mielenkiintoista ja kiinnostavaa testiautomaatioiden kirjoittaminen Robot Frameworkilla oli. Hyödynnettäessä avainsanapohjaista testiautomaatioviitekehystä testien kirjoittaminen oli tarvittavan tietopohjan saavuttamisen jälkeen mielekästä ja nopeaa. Myös oppimiskynnys Robotin kaltaisessa viiteautomaatiokehyksessä oli alhainen ja se rohkaisi kokeilemaan ja rakentamaan omia testejä. Pyrin pitämään yllä tämän työsuhteen aikana testiautomaatioon liittyviä oppeja ja tutustumaan aiheeseen lisää myös vapaa-ajalla. Uskon, että mielenkiinnostani testiautomaatioon ja siihen liittyvistä kokemuksista tulee varmasti olemaan hyötyä tulevissa työtehtävissä ja opinnoissa, liittyivät ne testaamiseen tai eivät.

Hyödyllisin jatkokehitysmahdollisuus tälle opinnäytetyölle on mielestäni käyttää sitä vertailukohtana uusia testiautomaatioon liittyviä tehtäviä silmällä pitäen. Pystyn tutkimaan uusien työtehtävien aikana tapahtunutta kehitystä ja oppimista vertaamalla sitä tämän opinnäytetyön aikana tapahtuneeseen kehitykseen. Vertailussa on kuitenkin huomioitava se, että tämän opinnäytetyön aikana tapahtunut kehitys on ollut suurta lähtötilanteen ollessa

hyvin erilainen. Uusi ja mielenkiintoinen tutkimusmahdollisuus olisi myös verrata, miten testiautomaatiota hyödynnetään julkisella sektorilla yksityisen sektorin sijasta. Näin olisi mahdollista tarkastella, täysin erilaisesta näkökulmasta, minkälaiseen testaukseen testiautomaatiota käytetään tai olisi mahdollista käyttää julkisella sektorilla. Vertailuun olisi mahdollista sisällyttää myös pohdintaa siitä, minkälaiset testiautomaatioviitekehykset sopisivat julkisen sektorin tarpeisiin parhaiten ja minkälaisia näkökulmia, saati tarvetta, julkisella sektorilla on yleisesti ottaen testiautomaatioon.

Pystyin hyödyntämään työni analysointia jatkuvasti kesätyösuhteeni aikana. Analysoimalla omaa työtäni pystyin tarkastelemaan kriittisestä näkökulmasta omia aikaansaannoksiani ja herättämään pohdintaa siitä, miten pystyisin kehittämään omaa toimintaani. Tämä auttoi taas haastamaan itseäni kokeilemaan ja etsimään uusia toimintatapoja. Tämän kesätyösuhteeni jälkeen pystyn hyödyntämään tekemääni analysointia myös tulevissa työtehtävissäni samaa toimintaperiaatetta hyödyntämällä.

5 Lähteet

Osa lähteistä on julkaistu kirjoittajan käyttämän käyttäjänimen alla.

Bradshaw, R. 2016. Give Your Automated Checks A Voice. Luettavissa: https://thefriendlytester.co.uk/2016/08/give-your-automated-checks-voice.html. Luettu: 24.6.2017.

Code Academy. 2017. Come help us build the education the world deserves. Luettavissa: https://www.codecademy.com/about. Luettu: 9.11.2017.

Computer Hope. 2017. HTML. Luettavissa: https://www.computerhope.com/jargon/h/html.htm. Luettu: 12.11.2017.

dvs1. 2017. Integrated Development Environments, The Python Wiki. Beaerton, US-OR, US: Python Software Foudation. Luettavissa: https://wiki.python.org/moin/IntegratedDevelopmentEnvironments. Luettu: 5.6.2017.

Edgren, R. 2012. The Little Black Book On Test Desing. Luettavissa: http://thetesteye.com/papers/TheLittleBlackBookOnTestDesign.pdf. Luettu: 2.7.2017.

Efficode. 2015. ImageHorizonLibrary. Luettavissa: http://eficode.github.io/robotframework-imagehorizonlibrary/doc/ImageHorizonLibrary.html. Luettu: 20.6.2017.

ExpressVPN. 2017. What is VPN?. Luettavissa: https://www.expressvpn.com/what-is-vpn. Luettu 12.11.2017.

Google Groups. 2016. 2 decimal points in robot framework. Luettavissa: https://groups.google.com/forum/#!topic/robotframework-users/NaNWhP73m_s. Luettu: 25.7.2017.

JetBrains. 2017a. PyCharm: Python IDE for Professional Developers by JetBrains. Luettavissa: https://www.jetbrains.com/pycharm/. Luettu: 5.6.2017.

JetBrains.2017b. Robot Framework Support. Luettavissa: https://plugins.jetbrains.com/plugin/7415-robot-framework-support. Luettu 9.11.2017.
Kaner, C., Bach, J. & Pettichord B. 2002. Lessons Learned In Software Testing. Wiley. New York. Luettu: 5.6.2017.

MatsWichmann. 2017. For Loops, The Python Wiki. Beaerton, US-OR, US: Python Software Foudation. Luettavissa: https://wiki.python.org/moin/ForLoop. Luettu: 21.6.2017.

Mozilla Foundation. 2017a. Firefox Add-ons. Luettavissa: https://addons.mozilla.org/en-US/firefox/addon/firebug/. Luettu: 10.11.2017.

Mozilla Foundation. 2017b. JavaScript Guide. Luettavissa: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide. Luettu 12.11.2017.

Page, A. 2013. The A Word. Lean Publishing. Luettu: 12.8.2017.

pekkaklarck, TehieWidget & charleswhchan. 2017. How to write good test cases using Robot Framework, GitHub. Luettavissa: https://github.com/robotframework/HowToWriteGoodTestCases/blob/master/HowToWriteGoodTestCases.rst#test-case-names. Luettu: 14.6.2017.

Robot Framework Foundation. 2017a. BuiltIn. Luettavissa: http://robotframework.org/robotframework/latest/libraries/BuiltIn.html. Luettu: 6.7.2017.

Robot Framework Foundation. 2017b. Generic test automation framework for acceptance testing. Luettavissa: www.robotframework.org/#introduction. Luettu: 5.6.2017.

Robot Framework Foundation. 2017c. Robot Framework User Guide Version 3.0.2. Luettavissa: http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html. Luettu: 5.6.2017.

Robot Framework Foundation. 2017d. Selenium2Library. Luettavissa: http://robotframework.org/Selenium2Library/Selenium2Library.html. Luettu: 21.6.2017.

Robot Framework Foundation. 2017e. String. Luettavissa: http://robotframework.org/robotframework/latest/libraries/String.html. Luettu: 8.7.2017.

Robot Framework Foundation. 2017f. The Docs. Luettavissa: http://robotframework.org/#documentation. Luettu 12.6.2017.

Scrum.org. 2017. What is a Product Backlog? Luettavissa: https://www.scrum.org/resources/what-is-a-product-backlog. Luettu: 27.6.2017.

Shatzer, L. 2016. Meet Jenkins, Jenkins Wiki. Luettavissa: https://wiki.jenkins.io/display/JENKINS/Meet+Jenkins. Luettu: 12.6.2017.

SmartBear Software. 2017. What Is Regression Testing? Luettavissa: https://smartbear.com/learn/automated-testing/what-is-regression-testing/. Luettu: 22.6.2017.

Stack Overflow. 2016. “Run Keyword If” and setting a variable. Luettavissa: https://stackoverflow.com/questions/35622902/run-keyword-if-and-setting-a-variable. Luettu 7.7.2017.

Stack Overflow. 2017. Rounding a number in python but keeping ending zeros. Luettavissa: https://stackoverflow.com/questions/19986662/rounding-a-number-in-python-but-keeping-ending-zeros. Luettu: 25.7.2017.

W3Schools. 2017a. The HTML DOM Document Object. Luettavissa: https://www.w3schools.com/jsref/dom_obj_document.asp. Luettu: 10.8.2017.

W3Schools. 2017b. XPath Syntax. Luettavissa: https://www.w3schools.com/xml/xpath_syntax.asp. Luettu: 21.7.2017.

W3Schools. 2017c. XPath Tutorial. Luettavissa: https://www.w3schools.com/xml/xpath_intro.asp. Luettu: 14.6.2017.

XMind. 2017. What Makes XMind Different? Luettavissa: http://www.xmind.net/features/. Luettu 27.6.2017.